Planification de personnel avec affectation de tâches fixées : méthodes et application dans un contexte médical

Tanguy Lapegue

Thèse de Doctorat

Tanguy LAPÈGUE

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'École nationale supérieure des mines de Nantes
sous le label de l'Université de Nantes Angers Le Mans

École doctorale : Sciences et technologies de l'information, et mathématiques

Discipline : Informatique et applications, section CNU 27
Spécialité : Recherche Opérationnelle
Unité de recherche : Institut de Recherche en Communications et Cybernétique de Nantes (IRCCyN)

Soutenue le 24 octobre 2014
Thèse n° : 2014EMNA0188

Planification de personnel avec affectation de tâches fixées : méthodes et application dans un contexte médical

JURY

Rapporteurs :	**M. Narendra JUSSIEN**, Professeur de l'Institut Mines-Telecom, directeur de Télécom Lille,
	M. Christian ARTIGUES, Directeur de recherche, Toulouse, LAAS-CNRS
Examinateurs :	**M. Éric HUMLER**, Directeur de la Recherche, École des Mines de Nantes
	M. Benoît ROTTEMBOURG, Directeur associé, Pricing & Revenue Management, Eurodecision
	M. Ameur SOUKHAL, Professeur des Universités, Polytech'Tours, LI
Invité :	**M. Nicolas FAUCHOUX**, Docteur, Biotrial
Directrice de thèse :	**M^me^ Christelle JUSSIEN-GUÉRET**, Professeur, Université d'Angers - LARIS
Co-encadrants :	**M^me^ Odile BELLENGUEZ-MORINEAU**, Chargée de recherche, École des Mines de Nantes - IRCCyN
	M. Damien PROT, Maître assistant associé, École des Mines de Nantes - IRCCyN

Notations générales

$\mathcal{D} = \{1, .., 7\}$	Ensemble des jours de la semaine
$\mathcal{N} = \{1, .., N\}$	Ensemble des employés disponibles
$\mathcal{T} = \{1, .., T\}$	Ensemble des tâches à assigner aux employés
$\mathcal{C} = \{\mathcal{K}_1, .., \mathcal{K}_C\}$	Ensemble des ensembles maximaux de tâches concomitantes entre elles
$\mathcal{N}^+ = \{1, .., N+1\}$	Ensemble des employés disponibles et de l'employé fictif
$\mathcal{M} = \{T+1, .., M\}$	Ensemble des tâches pré-assignées
$\mathcal{E} = \mathcal{T} \cup \mathcal{M}$	Ensemble des tâches à assigner et pré-assignées
$\forall \boldsymbol{n} \in \boldsymbol{\mathcal{N}}$,	**Pour tout employé $\boldsymbol{n}$:**
$\mathcal{T}_n \subseteq \mathcal{T}$	Ensemble des tâches assignables à n
$\mathcal{M}_n \subseteq \mathcal{M}$	Ensemble des tâches pré-assignées à n
$\mathcal{E}_n = \mathcal{T}_n \cup \mathcal{M}_n$	Ensemble des tâches assignables et pré-assignées à n
$\forall \boldsymbol{n} \in \boldsymbol{\mathcal{N}}, \forall \boldsymbol{d} \in \boldsymbol{\mathcal{D}}$,	**Pour tout employé $\boldsymbol{n}$, pour tout jour $\boldsymbol{d}$:**
$\mathcal{T}_{d,n} \subseteq \mathcal{T}_n$	Ensemble des tâches assignables à n le jour d
$\mathcal{M}_{d,n} \subseteq \mathcal{M}_n$	Ensemble des tâches pré-assignées à n le jour d
$\mathcal{E}_{d,n} = \mathcal{T}_{d,n} \cup \mathcal{M}_{d,n}$	Ensemble des tâches assignables et pré-assignées à n le jour d
$\forall \boldsymbol{n} \in \boldsymbol{\mathcal{N}}$,	**Pour tout employé $\boldsymbol{n}$:**
$target_n$	Objectif de travail médical de n (minutes)
off_n^{-1}	Ecart au dernier congé de n (jours)
$break_n^{-1}$	Ecart au dernier repos hebdomadaire de n (minutes)
$work_n^{-1}$	Ecart au dernier travail de n (minutes)
$\forall \boldsymbol{t} \in \boldsymbol{\mathcal{E}}$,	**Pour toute tâche $\boldsymbol{t}$:**
$start_t$	Date de début de t (minutes)
end_t	Date de fin de t (minutes)
$proc_t$	Durée de t (minutes)
day_t	Jour de travail de t

TABLE 1 – Notations générales

$\forall \boldsymbol{d} \in \boldsymbol{\mathcal{D}}$,	**Pour toute journée $\boldsymbol{d}$:**
$\textsc{DayS}_d$	Date de début (jour d à 6h = 360 + 1 440.d min)
$\textsc{DayE}_d$	Date de fin (jour d+1 à 6h = 360 + 1 440.(d+1) min)
$\textsc{LunchS}_d$	Début de la fenêtre horaire du déjeuner (jour d à 12h = 720 + 1 440.d min)
$\textsc{LunchE}_d$	Fin de la fenêtre horaire du déjeuner (jour d à 14h30 = 870 + 1 440.d min)
$\mathcal{L}_d^- \subseteq \mathcal{E}$	Tâches de la journée commençant avant $\textsc{LunchS}_d$
$\mathcal{L}_d^+ \subseteq \mathcal{E}$	Tâches de la journée terminant après $\textsc{LunchE}_d$
$\mathcal{O}_d \subseteq \mathcal{E}$	Tâches concomitantes à la journée
Législation	
$\textsc{Span}_{\textsc{max}}$	Temps présentiel journalier maximum (11h = 660 min)
$\textsc{WorkD}_{\textsc{max}}$	Temps travaillé journalier maximum (10h = 600 min)
$\textsc{WorkW}_{\textsc{max}}$	Temps travaillé hebdomadairement maximum (48h = 2 880 min)
$\textsc{BreakW}_{\textsc{min}}$	Repos hebdomadaire minimum (35h = 2 100 min)
$\textsc{BreakD}_{\textsc{min}}$	Repos journalier minimum (11h = 660 min)
LunchP	Durée minimum de la pause déjeuner (1h = 60 min)
Constantes	
DayP	Durée d'une journée (24h = 1 440 min)
SmallD	Durée minimum donnant accès à la pause déjeuner (5h = 300 min)
Week	Durée de l'horizon de planification (7j = 10 080 min)
α	Date de fin des journées vides (-1j = -1 440 min)
ω	Date de début des journées vides (8j = 11 520 min)

Table 2 – Notations générales (suite)

Remerciements

Je souhaite tout d'abord remercier Christelle Guéret, directrice de ma thèse, pour m'avoir fait confiance tout au long de ces trois années, pour sa gentillesse et ses encouragements. Je remercie également Narendra Jussien et Christian Artigues, les rapporteurs de ma thèse, pour leurs commentaires à la fois constructifs et bienveillants. Je voudrais également remercier tous mes examinateurs pour l'intérêt qu'ils ont porté à mes travaux, avec une mention spéciale pour Benoît Rottembourg dont les commentaires sont à la fois francs et chaleureux. À mon corps défendant, je crois bien qu'il restera connu dans la famille sous le pseudonyme de Monsieur Colonne :)

Je voudrais à présent remercier mes encadrants de thèse, Odile Bellenguez-Morineau et Damien Prot, pour ces trois années de travail commun. Grâce à leurs conseils et leurs encouragements, leur bonne humeur quotidienne et leur gentillesse, j'ai pu travailler dans des conditions idéales et je les en remercie. En dehors de ces trois années de travail intense, nous avons également sauvé le monde à de nombreuses reprises, parfois même avec des stratégies *basiques*, mangé jusqu'à 10 chamallows d'un coup (si seulement j'avais une photo...), usé le comptoir de la Cervoiserie, (re)découvert que les magrets au barbecue sont délicieux et même regardé un match de foot (*pour certains, c'est une habitude...*). Tous ces bons moments sont au final ce qu'il y a de plus important pour moi, merci.

Je tiens également à remercier chaleureusement les employés de Biotrial et plus particulièrement les infirmières planificatrices : Pascale Guezelou, Christèle Jourdan et Marie-Thérèse Cugnière ainsi que le docteur Nicolas Fauchoux. Malgré une charge de travail importante et une activité en pleine mutation, ils se sont beaucoup investis dans le projet MAESTRO ce qui était extrêmement motivant pour toute l'équipe. Grâce à une compréhension mutuelle des attentes de chacun, nous avons réussi à maintenir un bon compromis entre les activités de recherche et le développement de MAESTRO, ce qui nous a permis de conserver une bonne ambiance de travail tout au long du projet. J'espère de tout cœur que MAESTRO vous permettra rapidement de gérer plus sereinement votre activité si particulière. Par ailleurs, MAESTRO est le résultat d'un travail d'équipe et je voudrais également en profiter pour remercier tous ceux qui ont contribué au projet : Odile et Damien, bien sûr, mais également Mathieu, Christophe, Alexandre et Yassim.

Je voudrais également remercier toute l'équipe Systèmes Logistiques et de Production, et plus largement le personnel des Départements Automatique-Productique et Informatique. Je voudrais également remercier tous les joueurs de badminton, ainsi que tous ceux qui ont entretenu une bonne ambiance de travail en fêtant dignement chaque publication acceptée. Un remerciement spécial pour mes co-bureaux, Agnès Le Roux qui a supporté mes nombreuses jérémiades et qui a pris le temps de commenter la plupart des figures de ce document ainsi que Majid Eskandarpour qui nous a fourni en thé et en miel pendant au moins une année. Merci également à Jérémie Guerra pour cette très fameuse soirée à Bretignolles. Un grand merci également à Jean-Guillaume Fages avec qui j'ai partagé de bons moments de travail et de détente. Après avoir sauvé le monde et chassé moult zombis, nous comptons à présent chasser les clients, avec encore plus d'enthousiasme. Plus largement, je souhaiterais remercier tous les doctorants et post-doctorants, anciens comme nouveaux, pour la bonne ambiance de travail, l'entraide mutuelle et les discussions enrichissantes (ou pas). Pour finir, je voudrais remercier

ma famille pour leur soutien, leurs encouragements ainsi que pour les nombreux stages de remise en forme. Merci également à mon frère Keshik dont l'enthousiasme et le naturel sont toujours très rafraîchissants.

Merci à tous.

Table des matières

Liste des tableaux

Table des figures

Liste des Algorithmes

1

Introduction

1.1 Contexte de l'étude

La compétitivité des entreprises repose essentiellement sur leur capacité à utiliser efficacement les ressources dont elles disposent afin de satisfaire au mieux la demande de leurs clients. Selon le contexte, le terme « ressource » peut renvoyer à des notions très différentes telles que les matières premières, l'outil de production, les canaux de distribution ou le personnel. Nous nous intéressons ici à la gestion efficace du personnel qui est primordiale pour de nombreuses entreprises.

Bien que les problèmes de planification de personnel soient étudiés depuis plus de 60 ans, l'intérêt de la communauté scientifique à leur égard reste très vif encore aujourd'hui. Cela s'explique tout d'abord par la richesse et la grande diversité des problèmes théoriques sous-jacents, mais également par les innombrables applications pratiques. En effet, les problématiques de planification de personnel apparaissent dans des secteurs variés et touchent un grand nombre d'organisations. De plus, l'activité de chacune de ces organisations est assujettie à un contexte socio-économique bien spécifique et en perpétuelle mutation. Cela induit le besoin de développer des outils d'aide à la décision permettant aux décideurs d'appréhender plus rapidement et plus efficacement des données de plus en plus complexes pour dégager de nouvelles marges de productivité.

Avec l'accroissement des capacités de calcul des outils informatiques actuels, il devient possible d'envisager la résolution de problèmes plus difficiles et plus vastes qu'auparavant. Par conséquent, de plus en plus d'études abordent simultanément plusieurs aspects du problème, tels que l'attribution des jours de repos, la construction des horaires de travail et l'affectation des activités de travail aux employés. En raison de leur complexité théorique, ces problèmes requièrent des méthodes d'optimisation dédiées.

Dans cette thèse, nous nous focalisons sur le problème de planification de personnel de l'entreprise Biotrial qui est spécialisée dans la conduite de tests pharmaceutiques. De cette manière, nous pouvons évaluer l'intérêt pratique des méthodes proposées et, de son côté, Biotrial peut bénéficier des travaux de recherche réalisés durant cette thèse. Cet engagement réciproque se concrétise notamment par un financement de thèse de la part de Biotrial et, de notre côté, par la conception d'un outil d'aide à la décision, baptisé *MAESTRO* (***M****edical* ***A****ttendanc****E*** ***S****hit****T*** ***R****ostering* ***O****ptimisation*).

Le problème de planification de personnel de Biotrial survient dans le contexte très particulier des tests pharmaceutiques. Ces études permettent de vérifier que les médicaments développés par les entreprises pharmaceutiques sont bien tolérés par l'organisme humain, ce qui nécessite de nombreux contrôles, réalisés par des centres agréés par le Ministère de la Santé, tel que Biotrial. Afin de mener à bien les études qui leur sont confiées, ces centres de recherche recrutent des volontaires sains qui seront suivis durant toute la durée de l'étude de manière à vérifier les effets des nouveaux médicaments en toute sécurité. Chaque étude pharmaceutique se traduit par la réalisation d'actes médicaux sur des volontaires, ainsi que par de nombreuses analyses en laboratoire. L'organisation de cette activité particulière est entièrement régie par des protocoles de tests qui dressent la liste des tâches à réaliser ainsi que leurs horaires, ce qui permet de mesurer les effets du principe actif au fil du temps. Pour garantir le respect de chaque protocole de tests, Biotrial doit obligatoirement affecter chaque tâche à un employé précis, sachant que toutes les tâches non couvertes devront être attribuées à des intérimaires embauchées spécialement pour l'occasion. Il s'agit d'un problème complexe et récurrent puisqu'il faut faire le planning du personnel au moins une fois par semaine. Par ailleurs, les études étant toutes différentes les unes des autres, il est impossible de mettre en place un système de roulements du personnel, ce qui signifie qu'il faut également construire les horaires du personnel de manière à couvrir l'activité de l'entreprise qui s'exerce en continu. Pour organiser l'activité de ses employés, Biotrial mobilise deux infirmières planificatrices à temps plein. Ces infirmières maîtrisent complètement tous les aspects métiers du problème et ont de nombreuses années d'expériences, notamment en ce qui concerne la planification de l'activité du personnel. Malgré cela, la planification de l'activité du personnel reste une tâche pénible, longue, stressante et difficile à réaliser efficacement. Face à ce type de problème, les méthodes de résolution fondées sur des techniques issues de la recherche en intelligence artificielle et de la recherche opérationnelle ont fait la preuve de leur efficacité à de nombreuses reprises. Par exemple, la gestion des plannings du personnel en milieu hospitalier est un problème bien connu de la communauté scientifique et pour lequel de nombreuses méthodes de résolution ont déjà été développées et mises en application avec succès. Bien que le problème abordé ici soit particulièrement complexe et original, il semble par conséquent logique de chercher à le résoudre en mettant en œuvre ce genre de technologies.

1.2 Objectifs de la thèse

Cette thèse présente plusieurs objectifs majeurs, tant sur le plan académique que sur le plan industriel. D'un point de vue académique, il s'agit d'une part de caractériser scientifiquement le problème rencontré par notre partenaire industriel et de le positionner par rapport aux problèmes connexes de la littérature ; d'autre part, il s'agit également de proposer des méthodes de résolution innovantes et adaptées aux spécificités du problème. D'un point de vue industriel, il s'agit de développer un outil d'aide à la décision, reposant sur les méthodes de résolution développées durant cette thèse, de manière à faciliter, accélérer et optimiser la gestion quotidienne du personnel de Biotrial. Ce partenariat entre Biotrial et l'École des Mines de Nantes (EMN) permet ainsi de valoriser les travaux de recherche tout en dégageant un bénéfice réel pour Biotrial. Autrement dit, il s'agit d'initier un transfert technologique, ce qui s'inscrit dans la logique nationale de développement de l'innovation et de la compétitivité des entreprises, mais également dans une démarche de valorisation de la recherche.

1.3 Plan de la thèse

Le Chapitre 2 présente le cadre général de cette étude. Nous y introduisons tout d'abord le contexte métier de Biotrial ainsi que les différents problèmes rencontrés par notre partenaire. Nous déduisons de ces éléments que le problème de planification de personnel et d'affectation de tâches constitue un problème majeur dans le cadre de l'activité de Biotrial. Nous choisissons par conséquent de nous

focaliser sur ce problème particulier. Nous précisons alors le périmètre de l'étude en réalisant un certain nombre d'hypothèses de travail, puis nous discutons des données actuellement disponibles et concluons sur la nécessité de générer des instances de tests.

Le Chapitre 3 vise à positionner le problème retenu au Chapitre 2 par rapport à la littérature existante. Nous nous intéressons d'abord à la problématique de construction des horaires de travail du personnel avant de nous focaliser sur la problématique d'affectation de tâches fixées dans le temps. Pour chaque problème présenté, nous donnons une palette de travaux académiques, de manière à évaluer le périmètre couvert par le problème, puis nous concluons sur les points communs et les différences par rapport à notre cas d'étude. Plus précisément, nous nous intéressons tout d'abord au problème de planification du personnel en milieu hospitalier, en raison de la parenté des contextes métier, puis nous nous intéressons au problème de construction des horaires de travail dans un cadre plus général. Nous nous focalisons ensuite sur le cas particulier de la construction des horaires de travail du personnel navigant, en raison de la similitude entre la notion de trajet et celle de tâche non préemptive. Nous sortons ensuite du cadre de la gestion des ressources humaines, pour nous intéresser aux problèmes d'affectation de tâches fixées dans le temps. Nous concluons ce chapitre en soulignant les singularités du problème rencontré par notre partenaire.

Le Chapitre 4 introduit le cadre scientifique de cette étude. Nous donnons tout d'abord les notations utilisées à travers cette thèse, puis nous récapitulons les différentes contraintes du problème et proposons une modélisation mathématique de l'objectif d'équité retenu au Chapitre 2. À partir de ce cadre scientifique, nous donnons plusieurs algorithmes susceptibles d'intervenir lors de la résolution du problème. Nous proposons ensuite une modélisation sous forme d'un programme linéaire en nombres entiers que nous adaptons à plusieurs variantes du problème étudié. Ces différentes variantes abordent simultanément les problèmes de construction des horaires du personnel et d'affectation des tâches. L'intérêt de cette résolution simultanée est de construire des horaires de travail permettant a priori de mieux couvrir la charge de travail. En revanche, le fait de considérer le problème dans son ensemble conduit à un espace de recherche très vaste et compliqué à explorer efficacement. Une étude expérimentale met en évidence les limites d'une modélisation générale sous forme d'un programme linéaire en nombres entiers et nous conduit à considérer la conception d'approches plus adaptées.

Le Chapitre 5 propose une première méthode de résolution consistant à décomposer le problème en deux phases. La première phase permet de construire les horaires de travail du personnel, alors que la seconde phase vise à affecter un maximum de tâches aux employés, le plus équitablement possible. Nous itérons entre ces deux phases de manière à faire converger les horaires de travail en fonction des besoins de la seconde phase. Puisque chaque phase est résolue de manière heuristique, il est important de mettre en place des mécanismes de diversification de manière à trouver de bonnes solutions. Pour cela, nous utilisons deux stratégies différentes. En ce qui concerne la première phase, nous introduisons plusieurs opérateurs de voisinages partiellement aléatoires. En ce qui concerne la seconde phase, nous proposons un ensemble de critères heuristiques, conduisant chacun à une variante de la méthode, et nous appliquons régulièrement chacune de ces variantes. Une étude expérimentale permet tout d'abord de paramétrer chacune des deux phases ainsi que le fonctionnement global de la méthode en deux phases. Par la suite, nous retenons la meilleure configuration pour évaluer l'efficacité de cette méthode par rapport aux résultats obtenus au Chapitre 4.

Le Chapitre 6 reprend l'idée générale du Chapitre 4, selon laquelle une résolution simultanée des problèmes de construction d'horaires et d'affectation de tâches permet d'obtenir de meilleures solutions. Puisque la modélisation proposée au Chapitre 4 rencontre des difficultés à trouver de bonnes solutions, nous optons à présent pour une méthode de résolution à base de programmation par contraintes. Cette approche est en effet connue pour être efficace lorsqu'il s'agit de trouver une

bonne solution à un problème contraint. Nous proposons plusieurs stratégies de branchement permettant soit d'affecter un maximum de tâches aux employés, soit de minimiser l'inéquité entre les employés, ainsi que plusieurs stratégies d'exploration se focalisant soit sur les tâches difficiles à affecter, soit sur les tâches qui présentent un fort impact sur l'inéquité, soit sur l'impact d'une affectation vis-à-vis de l'horaire de travail correspondant. Ces différentes stratégies sont évaluées expérimentalement et les meilleurs résultats sont comparés à ceux obtenus par la méthode en deux phases.

Le Chapitre 7 propose une méthode de recherche à voisinage large, fondée sur le modèle du Chapitre 6. Cette méthode vise à explorer plus efficacement l'arbre de recherche en se concentrant sur certaines parties a priori susceptibles de conduire à des solutions améliorantes. Pour cela nous proposons deux voisinages orthogonaux : le premier se focalise sur la problématique d'affectation de tâches en considérant un sous-ensemble de tâches appartenant à un même créneau horaire ; le second se focalise sur la problématique de construction des horaires de travail du personnel en considérant un sous-ensemble d'employés sur tout l'horizon de planification. Par ailleurs, nous proposons également de gérer les problématiques d'affectation de tâches et de répartition équitable de la charge de travail selon des modalités différentes. En ce qui concerne l'affectation des tâches, nous cherchons des solutions améliorantes ou équivalentes. Pour obtenir des solutions équitables nous cherchons tout d'abord à diversifier les solutions obtenues en acceptant des solutions non améliorantes qui servent ensuite à construire des solutions plus équitables. Une étude expérimentale permet de conclure vis-à-vis de l'efficacité de cette méthode. Plus précisément, les résultats obtenus sont tout d'abord comparés à ceux du modèle de programmation par contraintes du Chapitre 6, puis à ceux de la méthode en deux phases présentée au Chapitre 5.

Le Chapitre 8 présente le projet MAESTRO qui fait le lien entre les travaux de recherche et les besoins opérationnels de Biotrial. Nous détaillons tout d'abord le mode de fonctionnement actuel de Biotrial et montrons les avantages potentiels d'un outil d'aide à la décision. Par la suite, nous présentons les caractéristiques techniques et fonctionnelles de l'outil développé pour Biotrial. Pour finir, nous revenons sur le déroulement global du projet et concluons sur la nécessité d'évaluer plus finement les gains obtenus au moyen de l'outil réalisé, de manière à le faire évoluer en fonction des besoins de notre partenaire.

Le Chapitre 9 s'éloigne quelque peu du problème opérationnel de notre partenaire industriel pour s'intéresser à un problème connexe de la littérature pour lequel les horaires du personnel sont déjà fixés et où l'objectif est de minimiser le nombre d'employés permettant de couvrir un ensemble de tâches fixées dans le temps. Bien que ce problème ne survienne pas directement dans le quotidien de Biotrial, il pourrait néanmoins intervenir lors du dimensionnement de l'effectif. Nous proposons une méthode de résolution fondée sur un modèle de programmation par contraintes ainsi que sur une stratégie de branchement tirant parti de la structure du problème et des caractéristiques de la méthode de résolution. Une étude expérimentale permet d'évaluer l'impact des différents composants de la méthode de résolution. Pour finir nous comparons les meilleurs résultats obtenus à ceux de la littérature.

Le Chapitre 10 présente un récapitulatif des travaux présentés dans cette thèse, ainsi que plusieurs pistes de recherche.

2

Présentation du problème

Nous nous intéressons dans ce chapitre à la formalisation du problème industriel rencontré par Biotrial. Nous présentons tout d'abord le contexte industriel ainsi que les notions clefs permettant de comprendre l'activité très particulière de Biotrial. Par la suite, nous présentons et justifions nos différentes hypothèses de travail avant de formaliser le problème étudié. Pour finir, nous présentons les différents jeux de données qui ont été utilisés tout au long de la thèse afin d'en valider les travaux.

Sommaire

2.1 Problème métier et contexte industriel

Les laboratoires pharmaceutiques qui souhaitent obtenir une autorisation de mise sur le marché pour un nouveau médicament doivent au préalable évaluer cliniquement tous ses effets sur le corps humain. Pour cela, il est nécessaire de réaliser une étude pharmaceutique auprès d'un organisme indépendant et agréé par le Ministère de la Santé, tel que Biotrial [29]. Ainsi, lorsqu'un laboratoire souhaite

réaliser une étude pharmaceutique, il transmet à Biotrial un protocole de tests décrivant en détail toutes les étapes permettant d'analyser précisément les effets du médicament. Ce protocole est tout d'abord étudié et validé par le médecin en charge de l'étude afin de déterminer précisément tous les examens médicaux à réaliser. Ces examens sont fixés dans le temps à la minute près, ce qui permet, par exemple, de mesurer l'évolution de la concentration de certaines substances dans le sang. Après validation du protocole de tests, Biotrial prend en charge la conduite de l'étude.

La toute première étape d'une étude pharmaceutique est de recruter un groupe de volontaires souhaitant participer à l'étude. Le service d'accueil de Biotrial se charge de recevoir les volontaires et de vérifier leur état de santé afin de détecter d'éventuelles contre-indications médicales qui les empêcheraient de participer à l'étude. Les volontaires admis sont indemnisés pour leur participation à l'étude. Une fois que le groupe de volontaires sains est complet, l'étude peut commencer. Le point de départ d'une étude correspond toujours à la phase d'administration : une partie des volontaires reçoit le médicament testé, alors que l'autre partie reçoit un placebo réalisé par le service pharmacie. Les études se déroulent en double aveugle ce qui signifie que ni les volontaires, ni les infirmiers ne connaissent la nature du produit administré (placebo ou médicament). Une série d'examens est alors réalisée sur chaque volontaire afin de surveiller tous les effets du médicament. Ces examens peuvent être pratiqués en ambulatoire ou bien requérir une hospitalisation. Dans tous les cas, Biotrial garantit une surveillance médicale adéquate, au sein d'un environnement cliniquement sécurisé. La plupart des études ne durent que quelques jours ou semaines, mais certaines études peuvent s'étaler sur plusieurs mois. De telles études servent typiquement à mesurer les effets d'un médicament sur le long terme. Dans ce cas, l'étude se compose d'une ou plusieurs visites médicales de quelques jours, effectuées plusieurs mois après la première administration du médicament.

Puisque la charge de travail relative aux études découle entièrement des protocoles de tests, il est crucial pour notre partenaire de gérer très finement son effectif de travail de manière à être capable de couvrir la charge de travail globale tout en maîtrisant les coûts en personnel et en respectant les contraintes organisationnelles et légales. Au fil de l'étude, les résultats cliniques sont systématiquement reportés sur un document de référence, retourné au laboratoire client à la fin de l'étude. La rédaction de cette synthèse requiert un temps de travail non négligeable. De cette activité particulière émergent plusieurs problèmes complexes portant sur différents niveaux temporels :

- D'un point de vue stratégique, il est nécessaire de dimensionner l'effectif requis en fonction de la charge de travail de l'entreprise. Ce processus de planification est réalisé sur un horizon de plusieurs mois ou années.

- D'un point de vue tactique, il est nécessaire de positionner les études pharmaceutiques de manière à ce que la charge de travail qui en résulte corresponde au mieux à la capacité de travail de l'entreprise. Ce processus de planification est réalisé sur un horizon de plusieurs semaines.

- D'un point de vue opérationnel, il est nécessaire de réaliser les plannings des employés de manière à disposer du personnel adéquat à la réalisation de tâches prévues. Il s'agit d'un processus complexe faisant intervenir la législation du travail et les contraintes relatives à l'organisation interne de l'entreprise, ainsi que les compétences, les disponibilités et les préférences des employés. Ce processus de planification du personnel est réalisé sur un horizon d'une semaine, en même temps que l'attribution des tâches aux employés. En cas de surcharge ponctuelle de travail, il peut être nécessaire de recourir à des intérimaires, ce qui revient à affiner les choix réalisés au niveau tactique et stratégique. Dans ce cas, l'objectif est de couvrir la charge de travail avec un nombre minimum d'intérimaires.

En ce qui concerne le dimensionnement de l'effectif, Biotrial a fait le choix de conserver un effectif minimum, quitte à recourir régulièrement à des intérimaires. De cette manière, les périodes de faible activité sont moins dommageables pour l'entreprise. Cette stratégie accentue en revanche la complexité des problématiques tactique et opérationnelle qui intègrent alors les prévisions d'embauches d'intérimaires.

Vis-à-vis du positionnement des études, Biotrial dispose en réalité d'une marge de manœuvre assez réduite. Les laboratoires clients imposent en effet de plus en plus fréquemment une date de début donnée. Lorsque le début de l'étude n'est pas imposé, il est souvent fortement contraint. Plus généralement, notons que la durée de vie des brevets, relativement courte par rapport au temps de développement de nouveaux médicaments, pousse les laboratoires pharmaceutiques à prendre des risques vis-à-vis de leur gestion de projets, afin de réduire les délais de validation. Par exemple, il arrive fréquemment que les laboratoires anticipent les autorisations de réalisation des tests cliniques et demandent à notre partenaire de planifier des tests pour lesquels l'autorisation n'a pas encore été obtenue. Cela peut conduire à annuler ou reporter des études entières à la dernière minute lorsque l'autorisation n'est pas délivrée à temps. Par conséquent, la gestion des intérimaires s'effectue presque exclusivement au niveau opérationnel.

Les plannings du personnel sont actuellement réalisés manuellement, ce qui mobilise deux décideurs très expérimentés à temps plein pour un effectif permanent d'une trentaine d'employés, mais pouvant doubler suite à une embauche massive d'intérimaires. En raison de la grande précision des protocoles de tests et de la complexité des activités de travail à organiser, la construction hebdomadaire des plannings est actuellement un processus très complexe. De plus, Biotrial a récemment entrepris des aménagements visant à accroître sa capacité d'accueil, notamment en doublant son nombre de lits, ce qui se traduira par un accroissement des données à prendre en compte au quotidien. Il est ainsi probable que les décideurs rencontrent de plus en plus de difficultés à appréhender toutes les données du problème, ce qui risque de conduire à une baisse de la qualité des plannings ainsi qu'à un accroissement important des besoins en intérimaires.

De nombreux échanges avec Biotrial nous ont permis de conclure que les problèmes de dimensionnement de l'effectif et de positionnement des études étaient moins adaptés à la mise en place d'un outil d'aide à la décision, notamment en raison de la faible marge de manœuvre. En revanche, le problème de gestion de l'effectif de travail semble tout à fait adapté à la mise en place d'un outil d'aide à la décision, ce qui nous a conduit à favoriser cette problématique.

2.1.1 Diversité des activités de travail

Au cours d'une même semaine, chaque employé peut être amené à réaliser des activités de nature très différentes, telles que des formations professionnelles, des réunions d'équipe, des analyses d'échantillons et bien entendu des actes cliniques (prise de sang, électrocardiogramme, *etc.*), pratiqués directement sur les volontaires, contrairement aux analyses d'échantillons qui se déroulent en laboratoire. Chaque étude est supervisée par une **équipe-projet** qui se réunit au début de l'étude pour valider le protocole de tests et construire le planning correspondant pour chaque volontaire. L'équipe-projet se charge également du bon déroulement de l'étude et de la rédaction du compte-rendu des résultats observés pour chaque volontaire. Ce compte-rendu est rempli au fil de l'étude par l'équipe-projet, ce qui requiert une part importante de leur temps de travail. Les activités du personnel peuvent se répartir en plusieurs catégories induisant chacune des contraintes de natures différentes. Nous avons donc choisi de distinguer trois types d'activités : les **activités administratives**, les **activités obligatoires** et les **activités médicales**.

Équipe-projet : il s'agit d'un groupe d'au moins trois employés en charge d'une étude donnée. Ce

groupe comprend systématiquement un technicien, un infirmier et un médecin. Le technicien et l'infirmier supervisent le déroulement des activités techniques et cliniques, et se chargent de rédiger les compte-rendus des résultats, alors que le médecin supervise le déroulement global de l'étude, valide le protocole de test et rédige le compte-rendu de sécurité et de tolérance du produit testé (son avis est décisif pour la poursuite ou non de l'étude).

Activités administratives : il s'agit notamment des activités de rédaction de compte-rendus. Ces activités peuvent être réalisées à tout moment de la semaine et ne requièrent aucune compétence spécifique, mais nécessitent en revanche une certaine expérience. Ces activités sont par conséquent confiées en priorité aux employés les plus expérimentés, jamais aux intérimaires. Plus précisément, ces activités sont généralement confiées aux équipe-projets. D'autre part, ces activités ne faisant pas partie du protocole de tests, elles peuvent être préemptées si besoin. Les employés organisent leur temps de travail administratif de manière autonome ce qui signifie que ces plages de travail ne sont pas indiquées explicitement sur le planning des employés.

Activités obligatoires : il s'agit des réunions d'équipes et des formations professionnelles. Ces tâches sont fixées dans le temps et pré-affectées à un employé donné en amont de la phase de planification du personnel. Les horaires de travail des employés doivent permettre aux employés de réaliser ces activités obligatoires, dans le respect des contraintes légales et organisationnelles.

Activités médicales : il s'agit des actes cliniques et techniques, requérant des compétences précises. Ces actes sont très majoritairement fixés dans le temps pour chaque volontaire concerné (date de début et durée), mais certains actes peuvent éventuellement être retardés, dans une certaine mesure, par rapport à leur date de début prévue. L'analyse de certains prélèvements, par exemple, peut être différée sans nuire à la fiabilité des résultats. Le fait de retarder certaines tâches est un procédé à éviter, destiné uniquement à gérer les pics d'activités. Contrairement aux activités obligatoires, les activités médicales ne sont pas pré-affectées à un employé donné et peuvent donc être réalisées par tout employé disponible et compétent, à condition bien sûr que toutes les contraintes légales et organisationnelles soient respectées. La gestion des activités médicales constitue le cœur du problème opérationnel de Biotrial, actuellement traité à la main, en même temps que la construction des horaires de travail des employés.

Pour chaque employé, la répartition du volume de travail total entre ces trois activités dépend des études pharmaceutiques en cours. Certaines études requièrent par exemple des compte-rendus très détaillés, ce qui accroît le temps de travail administratif. Notons également que le début de chaque nouvelle étude donne lieu à au moins une réunion de son équipe-projet afin de faire le point sur le protocole de tests de l'étude. Les périodes de travail correspondant au lancement de plusieurs études sont ainsi particulièrement chargées au niveau des activités obligatoires. Remarquons enfin que le nombre, la nature, ainsi que l'enchaînement des tâches médicales à réaliser varient entièrement d'une étude à une autre.

2.1.2 Particularités des activités médicales

Nous avons vu à la section précédente que les employés peuvent réaliser trois types d'activités : obligatoire, administrative et médicale. Nous avons également présenté les principales différences entre ces trois activités, ainsi que la manière dont ces activités sont gérées par le décideur (affectation, pré-affectation, préemption, *etc.*). Nous souhaitons à présent apporter quelques précisions sur les activités médicales dont la gestion constitue le cœur du problème étudié. Remarquons tout d'abord que la même notion *d'acte médical* peut renvoyer à différentes réalités selon le contexte et les interlocuteurs. Du point de vue des laboratoires pharmaceutiques, un acte médical correspond à la réalisation d'un acte précis sur un volontaire donné. On peut alors parler d'actes unitaires. Les décideurs visualisent le détail des actes unitaires sur un tableau Excel donnant les volontaires en colonne et les tâches

Day1									
Temps Théor. Absolu	Temps Théorique	Paramètres	Vol 1	Vol 2	Vol 3	Vol 4	Vol 5	Vol 6
D1		Evénements indésirables et traitements	NA					
	Prédose	Pose de cathéter	NA					
01:00		Petit déjeuner (30 min)	07:00	07:05	07:10	07:15	07:20	07:25
00:30		Heure de couchée (10 min)	07:30	07:35	07:40	07:45	07:50	07:55
00:10		Electrocardiogramme - ***Visite 6***	07:50	07:55	08:00	08:05	08:10	08:15
00:08		Electrocardiogramme - ***Visite 7***	07:52	07:57	08:02	08:07	08:12	08:17
00:06		Electrocardiogramme - ***Visite 8***	07:54	07:59	08:04	08:09	08:14	08:19
00:05		Pression artérielle + Température	07:55	08:00	08:05	08:10	08:15	08:20
00:02		Prélèvement sanguin (2 x 2ml, puis dans la glace)	07:58	08:03	08:08	08:13	08:18	08:23
00:00	H0	**Administration (avec 240ml d'eau)**	08:00	08:05	08:10	08:15	08:20	08:25

FIGURE 2.1 – Exemples d'actes médicaux

médicales en ligne (*cf.* Figure 2.1 pour une version simplifiée). Ainsi, chaque cellule correspond à un acte unitaire, comme par exemple : Electrocardiogramme - Visite 7, réalisé sur le volontaire n°2 à 7h57. Afin de respecter les protocoles de tests, les infirmiers et les techniciens en charge des actes médicaux doivent disposer du même niveau de détail que les laboratoires. En revanche, les décideurs en charge des plannings du personnel ne peuvent pas gérer un tel niveau de détail. Ils ont par conséquent pris l'habitude de regrouper certains actes médicaux entre eux, de manière à réduire le nombre de données à gérer. Ainsi, au lieu de gérer des actes unitaires (un acte sur un volontaire), les décideurs gèrent des actes multiples (plusieurs actes sur plusieurs volontaires). De cette manière, en affectant un seul acte multiple, le décideur peut affecter directement plusieurs actes unitaires. Il ne s'agit pas seulement d'une astuce pour réduire la quantité de données gérée par les décideurs mais surtout d'une pratique métier consistant à affecter une même personne sur une série d'actes médicaux communs à tout un groupe de volontaires. Bien sûr, les actes médicaux ne peuvent pas être regroupés n'importe comment et il faut une grande expérience pour déterminer les regroupements qui seront réalisables par les employés tout en minimisant les temps de battement entre deux actes unitaires. Remarquons en particulier que certains actes fixés au même horaire peuvent parfois être regroupés, car il est possible de les faire en même temps, alors que d'autres actes espacés de plusieurs minutes ne peuvent être regroupés, en raison d'une grande technicité par exemple. Cette distinction se fait sur la base d'une expertise métier très riche et en constante évolution puisqu'il est possible de modifier ce regroupement au cours d'une étude. D'un point de vue pratique, ce regroupement est réalisé sur le tableau Excel mentionné précédemment en attribuant une même couleur à plusieurs cellules. En reprenant l'exemple précédent, on peut ainsi voir que le décideur a regroupé tous les actes d'administration, de manière à ce qu'ils soient réalisés par un même employé (*cf.* Figure 2.2). De la même manière, les électrocardiogrammes, les pressions artérielles et les prélèvements sanguins des volontaires n°1, n°3 et n°5 ont été regroupés en un seul acte commençant à 7h30 et s'étalant sur 48 minutes.

2.1.3 Spécificités des employés

Le personnel de l'entreprise est réparti à travers quatre services : clinique, technique, accueil et pharmacie. Le service clinique se charge principalement des actes à réaliser sur les volontaires alors que le service technique s'occupe principalement des actes relatifs à l'analyse des prélèvements. Le service accueil gère l'arrivée des volontaires (inscription, bilan de santé, *etc.*) et pour finir, le service pharmacie est responsable de l'élaboration des préparations médicales destinées aux volontaires (médicament et placebo). Cette frontière entre les différents services repose essentiellement sur la maîtrise des compétences nécessaires à la réalisation des tâches affectées aux différents services. En

Day1									
Temps Théor. Absolu	**Temps Théorique**	**Paramètres**	**Vol 1**	**Vol 2**	**Vol 3**	**Vol 4**	**Vol 5**	**Vol 6**
D1		Evénements indésirables et traitements	NA					
	Prédose	Pose de cathéter	NA					
01:00		Petit déjeuner (30 min)	07:00	07:05	07:10	07:15	07:20	07:25
00:30		Heure de couchée (10 min)	07:30	07:35	07:40	07:45	07:50	07:55
00:10		Electrocardiogramme - ***Visite 6***	07:50	07:55	08:00	08:05	08:10	08:15
00:08		Electrocardiogramme - ***Visite 7***	07:52	07:57	08:02	08:07	08:12	08:17
00:06		Electrocardiogramme - ***Visite 8***	07:54	07:59	08:04	08:09	08:14	08:19
00:05		Pression artérielle + Température	07:55	08:00	08:05	08:10	08:15	08:20
00:02		Prélèvement sanguin *(2 x 2ml, puis dans la glace)*	07:58	08:03	08:08	08:13	08:18	08:23
00:00	**H0**	**Administration (avec 240ml d'eau)**	08:00	08:05	08:10	08:15	08:20	08:25

FIGURE 2.2 – Exemples de regroupements d'actes médicaux

cas de besoin, il est cependant possible de réaliser des transferts d'employés d'un service à l'autre. Par exemple, les services cliniques et techniques peuvent s'échanger du personnel en cas de surcharge de travail de l'un ou l'autre des services, ce qui en pratique est très fréquent. En revanche, pour des raisons de respect du protocole, les employés du service pharmacie ayant participé à la préparation d'une étude donnée ne peuvent réaliser aucune des tâches cliniques ou techniques de cette étude puisqu'ils savent quels volontaires reçoivent le médicament et quels volontaires reçoivent le placebo. On parle alors d'étude « interdite » pour signifier qu'il est impossible d'assigner des tâches de cette étude à ces employés. Il leur est cependant possible d'assister le service accueil en cas de besoin. De manière générale, les employés préfèrent réaliser les tâches relatives à leur propre service.

2.1.4 Contraintes légales et organisationnelles

L'ensemble des contraintes à prendre en compte lors de l'élaboration des plannings hebdomadaires comprend d'une part les contraintes légales et d'autre part les contraintes organisationnelles, *i.e.* les contraintes relatives au fonctionnement interne de l'entreprise.

Contraintes légales

La distribution des tâches étant très variable d'un jour à l'autre et d'une semaine à une autre, les horaires du personnel ne sont pas fixés à l'avance. La charge de travail médical de Biotrial est tellement variable qu'il est impossible de prévoir des roulements ou des horaires réguliers. Par conséquent, les plannings sont entièrement contruits chaque semaine en fonction de l'activité. Il est donc nécessaire de vérifier de nombreuses contraintes légales relatives au temps de repos et au temps de travail. Ces contraintes permettent de garantir que les plannings construits respectent la législation en vigueur au sein de l'entreprise. Toutes ces contraintes reposent sur quelques notions clés que nous identifions sous les termes suivants : **vacation**, **temps présentiel**, **temps travaillé** et **jour de repos**.

Vacation : il s'agit d'un horaire de travail journalier, attribué à un employé. Cet horaire peut comprendre des activités médicales et/ou obligatoires. Les activités administratives ainsi que les pauses n'apparaissent pas explicitement sur les vacations. Autrement dit, une vacation correspond à un horaire de présence minimum, pouvant être étendu au début ou à la fin, au gré des employés mais sous réserve de respecter les contraintes légales, de manière à réaliser leurs activités administratives. Pour cette raison, les vacations n'ont pas de durée minimale.

Temps présentiel : il s'agit de l'amplitude horaire d'une vacation. Le temps présentiel d'un employé travaillant de 8h à 16h est donc de 8h.

Temps travaillé : il s'agit du volume horaire travaillé durant une vacation. Le temps travaillé d'un employé présent de 8h à 16h avec une pause déjeuner d'une heure est donc de 7h.

Jour de repos : il s'agit d'un intervalle horaire entièrement non travaillé d'une durée de 24h. Dans l'intérêt du salarié, il est généralement préférable de placer ce repos le dimanche, mais cela n'est pas toujours possible, puisque l'activité de Biotrial est continue.

À partir de ces différentes notions, il est possible d'énoncer les principales contraintes légales du problème. Tout d'abord, le temps présentiel quotidien doit être inférieur à 11 heures et le temps travaillé quotidiennement ne doit pas dépasser 10 heures. De plus, les employés travaillant le matin bénéficient d'une pause de 18 minutes et d'une pause déjeuner d'une heure entre 12h et 14h30. Ces deux pauses sont décomptées du temps de travail, *i.e.* le temps travaillé d'un tel employé correspond au temps présentiel diminué de 1h18. D'autre part, les employés ne travaillant pas le matin mais travaillant sur plus de 6h bénéficient d'une pause de 30 minutes comptée dans le temps de travail, *i.e.* le temps travaillé d'un tel employé correspond à son temps présentiel. Remarquons dès à présent que le respect de ces pauses peut être problématique, puisque certaines activités médicales durent plusieurs heures. Ainsi, par exemple, un employé étant affecté à une activité médicale commençant à 10h et se terminant à 15h, ne peut pas prendre une pause déjeuner « classique ». Ces contraintes sont donc considérées comme souples et les conflits sont arbitrés par l'infirmière en chef afin que les vacations des employés restent supportables, même si certaines « contraintes » sont violées. D'autres contraintes portent sur le temps de travail hebdomadaire des employés. Ainsi, le temps travaillé hebdomadairement ne doit pas dépasser 48 heures. Enfin, certaines contraintes glissantes sont également à prendre en compte. Par exemple, une période de repos hebdomadaire d'une durée minimale de 35 heures, ainsi qu'au moins un jour de repos, doivent être observés sur une période de 7 jours consécutifs. D'autre part, deux vacations consécutives doivent être séparées par un temps de repos minimum de 11 heures. Notons que le respect des contraintes glissantes nécessite de prendre en compte le planning de la semaine passée, afin de vérifier que le temps de repos entre la vacation du dimanche et celle du lundi est bien respecté, mais également que le jour de repos ainsi que le repos hebdomadaire de la semaine courante surviennent suffisamment tôt pour que la contrainte soit respectée, selon le placement du repos précédent.

Contraintes organisationnelles

Les horaires de travail d'un même employé peuvent être très différents d'un jour à l'autre de la semaine, car ils dépendent des tâches affectées à cet employé. En théorie, n'importe quel horaire de travail respectant les contraintes relatives à la législation du travail peut être appliqué. En pratique, les horaires portant à la fois sur la nuit et sur le matin sont jugés trop pénibles pour être appliqués. Ainsi, un même employé ne peut être affecté à deux tâches commençant par exemple à 5h et 8h le mardi matin car la première tâche est considérée par l'infirmière en chef comme une tâche de nuit, alors que la seconde est considérée comme une tâche de matin. Remarquons que certaines tâches peuvent commencer la nuit et finir le matin. Dans ce cas, l'infirmière en chef considère ces tâches comme étant des tâches de nuit, car les infirmières travaillant la nuit préfèrent souvent réaliser des vacations plus longues mais moins nombreuses.

Le respect des protocoles de tests entraîne de nombreuses contraintes organisationnelles. Tout d'abord, chaque tâche doit être affectée à exactement un employé maîtrisant toutes les compétences nécessaires à la réalisation de cette tâche. De plus, les dates de début des tâches doivent être scrupuleusement respectées. D'autre part, les employés affectés à une tâche doivent être disponibles sur toute la durée de cette tâche, ce qui nécessite de prendre en compte les indisponibilités des employés (congés, réunions, *etc.*). De plus, un même employé ne peut être affecté à deux tâches concomitantes. Afin de garantir la fiabilité des résultats, les infirmières ayant participé à l'élaboration des préparations destinées aux volontaires ne peuvent être affectées à aucune tâche médicale.

2.1.5 Appréciation d'un planning

Un planning est considéré comme réalisable s'il respecte toutes les contraintes légales et organisationnelles de l'entreprise. Plus particulièrement, il est nécessaire d'obtenir une affectation complète des tâches aux employés. Il s'agit par conséquent de regrouper les tâches en horaires de travail et de les affecter aux employés adéquats, *i.e.* compétents et disponibles, de manière à ce que les plannings individuels qui en résultent respectent les différentes contraintes. Une illustration de ce problème sur un horizon d'une journée est donnée à la Figure 2.3.

La qualité d'un planning réalisable dépend de la charge de travail globale des employés. Plus précisément, il s'agit de répartir équitablement la charge de travail globale, de manière à éviter que certains employés soient en surcharge alors que d'autres seraient en sous-charge. Afin de prendre en compte toutes les activités réalisées par les employés, cette charge de travail globale inclut la **charge médicale**, la **charge obligatoire** et la **charge administrative**. Puisque les activités administratives n'apparaissent pas sur le planning du personnel, afin de laisser la liberté aux employés d'organiser ce temps de travail, l'infirmière en chef ne peut pas vérifier directement sur le planning la charge de travail globale des employés. Elle doit donc garder à l'esprit cette charge administrative, différente pour chaque employé, de manière à solliciter davantage les employés qui ont une faible charge administrative.

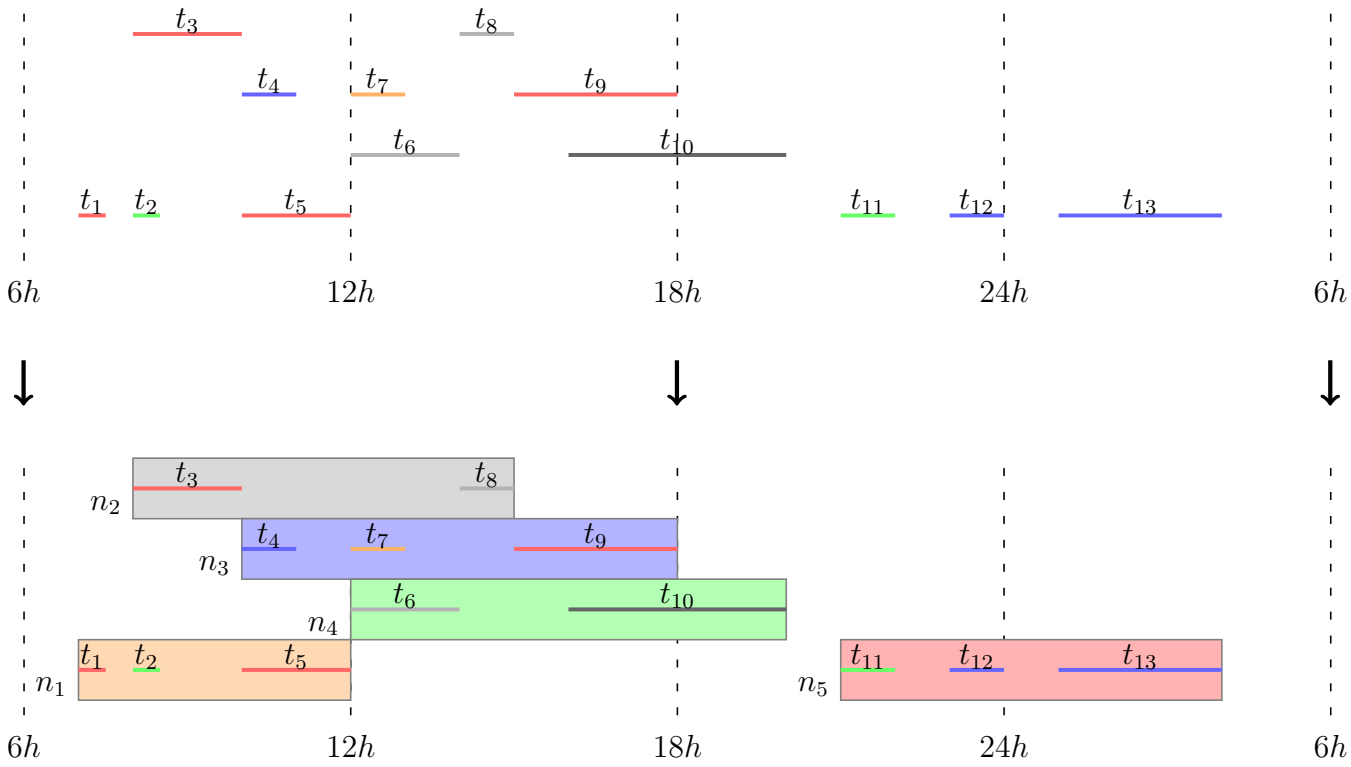


FIGURE 2.3 – Exemple du problème sur un jour, avec 13 tâches $(t_1, .., t_{13})$ et 5 employés $(n_1, .., n_5)$.

Charge médicale : Il s'agit du temps d'exécution cumulé des différentes tâches médicales affectées à un employé donné. La charge médicale d'un employé réalisant une tâche clinique de 8h à 9h, participant à une réunion de 13h à 14h et réalisant une tâche technique de 15h à 16h est donc de 2h (la durée des tâches cliniques et techniques, mais pas celle de la réunion).

Charge obligatoire : Il s'agit du temps d'exécution cumulé des différentes tâches obligatoires affectées à un employé donné. En reprenant l'exemple précédent, la charge obligatoire est donc de 1h.

Charge administrative : Il s'agit du temps a priori nécessaire à la réalisation des différentes tâches administratives d'un employé. Cette charge est estimée par l'infirmière en chef. Puisqu'il est difficile de réaliser un suivi précis des activités administratives, celles-ci n'apparaissent pas sur le planning du personnel.

2.2 Hypothèses de travail

L'entreprise estimant que certaines données ne peuvent être formalisées parfaitement, nous avons décidé de nous concentrer sur le cœur du problème industriel afin de répondre au mieux aux besoins opérationnels. Cela nous a conduits à exclure certains cas particuliers que l'entreprise traitera soit en amont par une réorganisation, soit en aval par un ajustement du planning au cas par cas, en prenant en compte une expertise métier parfois très difficile à généraliser. L'outil d'aide à la décision développé pour Biotrial prendra cependant en compte un maximum de données et de critères métier (*cf.* Chapitre 8).

2.2.1 Simplification des données

Sur les quatre services du personnel (accueil, pharmacie, clinique et technique, *cf.* Section 2.1.3, p. 25), nous avons décidé d'un commun accord avec l'entreprise de focaliser notre étude sur les services clinique et technique. Ces deux services sont en effet en charge des activités médicales qui requièrent une attention particulière, notamment en raison de la précision des tâches à attribuer et des grandes fluctuations d'activité affectant ces services. Le service pharmacie n'étant pas pris en compte, il devient alors inutile de gérer les études interdites (*cf.* Section 2.1.3 p. 25). Puisque le fait de retarder certaines tâches est un procédé à éviter, nous considérons ici que toutes les tâches doivent être planifiées à leur date de début prévue. La durée et la date de début de toutes les tâches médicales sont ainsi fixées dans le temps avec une précision d'une minute afin de respecter les protocoles établis par les laboratoires. Si besoin, les tâches qui auraient pu être retardées pourront être déplacées manuellement par le décideur.

2.2.2 Gestion des contraintes

En ce qui concerne les pauses de 18 et 30 minutes, nous avons décidé de ne pas les garantir car nous avons estimé avec l'infirmière en chef que ces pauses seraient automatiquement respectées dans la quasi totalité des cas, en raison des dates de début des tâches, fixées dans le temps. Il est en effet très peu probable d'arriver à produire des plannings ne présentant aucun temps d'attente entre les différentes tâches affectées à un même employé. De plus, même les plannings qui ne respectent pas ces temps de pause peuvent finalement être acceptés si besoin.

Vis-à-vis des pauses déjeuner, afin de pallier certains cas pathologiques (*cf.* Section 2.1.4), nous avons décidé de les prendre en compte de manière assouplie : les vacations de plus de 5 heures, commençant avant 12h et terminant après 14h30 donnent accès à un temps de repos d'une heure pouvant éventuellement être morcelé. Cette durée limite de 5 heures s'explique en deux temps. Premièrement, nous avons estimé qu'en-dessous de 5 heures, il n'était pas nécessaire de garantir un temps de pause, celui-ci pouvant être pris en fin de vacation. Encore une fois, il s'agit d'un arrangement pouvant également être appliqué en pratique. Deuxièmement, les tâches médicales les plus longues durant jusqu'à 5 heures, il ne serait pas possible de respecter le temps de pause des

employés affectés à ces tâches sans compléter leur vacation, déjà chargée, en y ajoutant une autre tâche. Considérons en effet un employé affecté à une seule tâche commençant à 10h et se terminant à 15h. Si on souhaite que cet employé bénéficie d'une heure de pause déjeuner, il faut alors lui affecter une autre tâche se terminant à 9h ou commençant à 16h. Cela conduirait cependant à une vacation encore plus pénible que la vacation initiale de 10 à 15 heures. Le fait d'autoriser les pauses morcelées permet de mieux couvrir la charge de travail, ce qui reste la préoccupation principale de l'entreprise. D'autre part, le respect de la pause déjeuner entraîne de nombreux cas pathologiques pour lesquels il n'existe pas vraiment de solution idéale, comme cela a été signalé à la Section 2.1.4.

2.2.3 Critères d'appréciation d'un planning

L'une des contraintes principales du problème industriel étant d'affecter toutes les tâches au personnel disponible, il semblerait naturel de considérer l'affectation des tâches comme une contrainte inviolable. En pratique, il peut cependant arriver qu'il soit impossible de trouver une telle affectation. L'entreprise complète alors son effectif au moyen d'intérimaires afin de réaliser les tâches restantes. Biotrial ayant fait le choix stratégique de travailler avec un effectif permanent minimum, la gestion de ces pics d'activité peut la conduire, dans les cas extrêmes, à doubler son effectif de travail en engageant des intérimaires. Remarquons que le recours aux intérimaires est préjudiciable à l'entreprise à plus d'un titre. En effet, outre les coûts financiers, le recours aux intérimaires conduit à des problèmes de maîtrise des compétences demandées, ce qui requiert de planifier des formations en doublon. D'autre part, les intérimaires ont parfois du mal à s'adapter au rythme de l'entreprise, ce qui peut entraîner des départs impromptus, ou des périodes d'adaptation relativement peu productives.

Après plusieurs échanges avec le décideur il est apparu que ce dernier souhaitait conserver la main sur l'utilisation des intérimaires. Il est en effet difficile de définir un profil type de compétences pour les intérimaires, car leurs compétences ainsi que leur aptitude à s'adapter au rythme soutenu de l'entreprise varient énormément. De plus, il est difficile de construire un horaire de travail type pour les intérimaires, car la durée de leur embauche peut aller de la demie journée à plusieurs semaines, selon les besoins. Par ailleurs, selon la complexité de l'activité, il peut être intéressant d'embaucher un intérimaire sur une période plus longue que nécessaire, de manière à capitaliser sur sa formation. Pour autant, lorsqu'il est impossible d'affecter toutes les tâches, le décideur souhaite qu'on lui propose un planning initial, de manière à pouvoir évaluer les besoins minimum en intérimaires. Ce planning initial doit être construit de manière à apporter au décideur un maximum d'information sur les besoins en intérimaires. Idéalement, il s'agirait de construire un planning dont l'effectif de travail pourrait être complété par un nombre minimum d'intérimaires, sans qu'il y ait besoin de modifier les affectations des autres employés. Le profil de compétences ainsi que l'horizon de recrutement des intérimaires étant inconnus, la construction d'un tel planning semble cependant particulièrement difficile. Considérons par exemple la minimisation du nombre de vacations permettant de couvrir les tâches restantes. Cet objectif présente l'avantage de prendre en compte le positionnement temporel des tâches, permettant ainsi de gérer certains cas pathologiques. Par exemple, si trois tâches ne peuvent être affectées, cet objectif favorisera les solutions pour lesquelles ces trois tâches sont suffisamment proches pour être regroupées dans une même vacation, sans pour autant être concomitantes, de manière à pouvoir être affectées à un unique intérimaire. L'inconvénient de cet objectif est qu'il ne tient pas compte des compétences requises pour réaliser les tâches restantes, pouvant ainsi favoriser des solutions pour lesquelles les tâches restantes requièrent des compétences rares, non maîtrisées par les intérimaires. Si l'objectif est de minimiser le nombre et la rareté des compétences requises pour réaliser les tâches non attribuées, il sera alors plus facile d'affecter les tâches restantes aux intérimaires, mais il est possible que ces tâches soient éparpillées sur toute la semaine, ou au contraire concomitantes, ce qui conduirait à surestimer soit le nombre d'intérimaires, soit le temps

pendant lequel mobiliser des intérimaires. Une autre option pourrait être de minimiser la charge médicale non attribuée, de manière à ce que la charge à réaffecter soit faible. Cela conduirait à favoriser l'attribution des tâches les plus longues au détriment des tâches les plus courtes. Un tel objectif risquerait donc d'accroître le nombre de tâches non attribuées, augmentant potentiellement le nombre de compétences différentes associées aux tâches non affectées. Ces différents exemples montrent qu'il est difficile de concevoir une stratégie générale permettant de construire un planning dont l'effectif de travail pourrait être complété par un nombre minimum d'intérimaires, sans qu'il y ait besoin de modifier les affectations des autres employés. Par conséquent, nous optons ici pour une autre stratégie consistant à construire un planning facilement interprétable par le décideur. Afin de réduire la quantité d'information à prendre en compte, nous cherchons à minimiser le nombre de tâches non affectées. Le décideur peut alors réaliser des arbitrages sur les quelques tâches restantes de manière à trouver une solution complète, ou bien ajouter des intérimaires et replanifier, de manière à obtenir un planning satisfaisant au bout de quelques itérations. Notons que les intérimaires pourront être mobilisés sur une ou plusieurs plages horaires, en fonction des besoins, et que le décideur pourra également positionner les plages de formation nécessaires. De cette manière, un planning pour lequel toutes les tâches sont affectées est toujours préféré à un planning pour lequel certaines tâches ne sont pas affectées, mais pour autant, en cas d'irréalisabilité, le décideur bénéficie d'une base de départ pour évaluer les besoins en intérimaires.

Lorsque toutes les tâches sont affectées, il convient de répartir équitablement la charge de travail globale entre les employés, ce qui requiert de prendre en compte les activités médicales, obligatoires et administratives (*cf.* Section 2.1.5). Pour répartir équitablement la charge de travail globale, les infirmières en chef prennent en compte les activités administratives de manière informelle. Notre partenaire souhaite conserver ce fonctionnement, pour éviter de contraindre davantage les horaires de travail des employés en positionnant leur travail administratif. Cela nous a conduit à définir pour chaque employé une **charge médicale idéale** qui interviendra dans la répartition équitable de la charge de travail médicale. Pour un employé donné, cette charge idéale est d'autant plus grande que les charges administrative et obligatoire sont faibles sur la période considérée. Cette notion permet de traduire simplement le fait qu'un employé fortement chargé en activités administratives et obligatoires ne doit pas être trop chargé en activités médicales. Nous capturons ainsi ce que le décideur prend en compte de manière informelle lors de la construction du planning. À partir de cette notion de charge idéale de travail médical, il est possible de définir les notions de **surcharge médicale** et de **sous-charge médicale**. Plus un employé est en surcharge médicale, moins il aura de temps pour réaliser le reste de ses activités. À l'inverse, plus un employé est en sous-charge médicale, plus il aura de temps pour réaliser le reste de ses activités. Par conséquent, la répartition de la charge de travail globale est d'autant plus équitable que les employés ayant la plus grande surcharge médicale et ceux ayant la plus grande sous-charge médicale sont proches de leur charge idéale. Ainsi, par exemple, une solution où tous les employés respectent leur charge médicale idéale correspond à une répartition idéale du travail. De la même manière, si tous les employés sont en surcharge médicale à hauteur de deux heures par exemple, cela correspond également à une répartition idéale de la charge de travail, car cela signifie que la surcharge globale d'activité est équitablement répartie entre les employés. À l'inverse, si certains employés sont en surcharge médicale d'une heure, alors que d'autres sont en sous-charge médicale d'une heure, alors cela n'est pas équitable. Autrement dit, nous considérons que la répartition de la charge globale est d'autant plus équitable que la différence de traitement entre les employés les plus pénalisés et les employés les moins pénalisés est faible.

Charge médicale idéale : il s'agit d'un volume de travail, donné en minutes. Ce volume dépend de la charge administrative de l'employé correspondant. Ainsi, plus un employé à une charge administrative importante, plus sa charge médicale idéale est faible. Il s'agit donc d'une donnée du problème, définie par le décideur, de manière à estimer les proportions à respecter entre la charge médicale et les charges administratives et obligatoires pour chaque employé. Il ne s'agit

donc pas d'une valeur maximale.

Sur/Sous charge médicale : se dit de l'activité médicale d'un employé dont la charge de travail médical affectée est supérieure/inférieure à sa charge médicale idéale.

2.2.4 Périmètre du problème

Nous résumons ici les différentes hypothèses de travail établies avec notre partenaire afin d'uniformiser les données tout en gardant à l'esprit que les cas spécifiques seront gérés par le décideur :

R 1: seuls les services clinique et technique sont pris en compte ;
R 2: les études interdites ne sont pas prises en compte de façon spécifique ;
R 3: les tâches sont toutes considérées comme étant entièrement fixées dans le temps ;
R 4: les pauses de 18 et 30 minutes ne sont pas prises explicitement en compte ;
R 5: les pauses déjeuner sont vérifiées pour les vacations de plus de 5h et peuvent être morcelées.

Ces restrictions portent à la fois sur les données d'entrées et les contraintes du problème. Notons que la restriction R 2 n'impacte pas la modélisation du problème puisque les études interdites peuvent être aisément prises en compte via des compétences particulières. En revanche, la restriction R 3 impacte fortement la modélisation du problème puisqu'elle permet de s'affranchir complétement du positionnement des tâches. Conceptuellement, il s'agit de la restriction la plus forte réalisée dans le cadre de l'étude. Cependant, cette restriction n'a que peu d'impact sur la qualité des plannings puisque les tâches pouvant être retardées sont assez rares et qu'il est préférable de ne pas les retarder. Les restrictions R 4 et R 5 s'expliquent par les temps de battement entre les tâches, ainsi que par la possibilité de réaliser des arbitrages reposant sur une expertise métier impossible à saisir parfaitement.

2.3 Données de test

Au regard des nombreuses spécificités du problème, il ne nous semble pas possible de trouver dans la littérature des jeux de tests suffisamment proches de notre cas d'étude pour être pertinents. Le problème étudié étant issu d'une étude industrielle, il est cependant crucial de vérifier le comportement des méthodes de résolution sur des données susceptibles de correspondre à la réalité de l'entreprise. Étant donné que l'entreprise partenaire ne pouvait pas nous fournir de données réelles comprenant les différents éléments de décision, nous proposons par conséquent un nouveau jeu de données générées de manière à correspondre aux données industrielles. Il est regrettable que nous n'ayons pu bénéficier de données réelles, mais cela s'explique par le fait que tout le processus de planification du personnel est réalisé à la main par les infirmières en chef, ce qui requiert d'ailleurs une très grande expertise métier.

2.3.1 Caractéristiques des données industrielles

Une semaine classique correspond environ à 200 tâches dont la durée varie entre quelques minutes et plusieurs heures. Ces tâches sont distribuées sur toute la semaine et peuvent commencer à toute heure du jour ou de la nuit. Les tâches à réaliser le week-end ou durant la nuit sont cependant beaucoup plus rares alors qu'il est fréquent d'observer un pic d'activité à 8h. En tout, plus d'une trentaine d'actes cliniques différents sont répertoriés (analyses médicales, prises de sang, pipetages, électrocardiogrammes, encéphalogrammes, pressions artérielles, *etc.*). En temps normal, l'entreprise dispose d'une trentaine d'employés très polyvalents pour suivre les protocoles de tests. Chaque em-

ployé présente des indisponibilités non travaillées (congés posés, absences, contrats particuliers, *etc.*) ainsi que des indisponibilités travaillées (formations et réunions).

2.3.2 Générations de jeux d'instances

Étant donné que la création des plannings du personnel repose actuellement sur l'expérience de quelques décideurs qui conservent peu de traces des contraintes considérées au niveau opérationnel, nous n'avons pas pu bénéficier de jeux de données sous un format directement exploitable. En consultant d'anciens plannings, nous avons cependant pu estimer la charge de travail moyenne des employés, ainsi que la distribution globale de la charge de travail. À partir de ces observations, nous avons généré plusieurs instances correspondant à une semaine de travail en fonction des paramètres I_t, $I_v$et I_c, correspondants respectivement au nombre de tâches, à la charge de travail moyenne, et à la distribution de compétences. Les caractéristiques principales de ces paramètres sont récapitulées à la Table 2.1. En croisant toutes les configurations de ces différents paramètres, nous obtenons 24 jeux d'instances. De manière à pouvoir étudier chaque jeu d'instances, nous avons généré 30 instances de chaque configuration, soit un total de 720 instances, disponibles en téléchargement [167].

I_t : nombre de tâches cliniques sur une semaine. Ce paramètre permet de faire varier la taille des instances étudiées, ce qui nous permet d'évaluer le comportement des méthodes de résolution face à une instance de taille classique, mais également d'évaluer leurs capacités de passage à l'échelle. Nous proposons d'étudier des jeux de données avec 100, 200, 300 et 400 tâches cliniques. Nous estimons en effet qu'une instance classique comprend environ 200 tâches cliniques.

I_v : volume de travail clinique moyen des employés sur une semaine, en minutes. Un volume de travail moyen classique correspond à une valeur de 800 minutes. Par conséquent, nous proposons d'étudier trois configurations particulières : 600 minutes (faible), 800 minutes (classique) et 1 000 minutes (important). Ces trois configurations nous permettent de vérifier le comportement des méthodes en cas de sous-charge, de surcharge ou d'équilibre entre la capacité de travail et la demande.

I_c : répartition des compétences au sein de l'effectif disponible. Nous proposons d'étudier deux configurations différentes. La première configuration, notée I_{cc}, correspond à une répartition homogène des compétences, ce qui nous permet de prendre en compte le cas pratique où toutes les compétences requises sont jugées « communes », *i.e.* maîtrisées par un grand nombre d'employés. La seconde configuration, notée I_{cr}, complète la première configuration en y ajoutant la notion de « compétences rares », ce qui arrive lors de certaines études requérant des compétences particulières pour lesquelles peu d'employés sont formés.

Attribut	Valeurs
Nombre de tâches (I_t)	{ 100, 200, 300, 400 }
Nombre d'employés	$[\![15, 50]\!]$
Charge médicale moyenne par employé (I_v)	{ 600, 800, 1 000 }
Distribution des compétences (I_c)	{I_{cc}, I_{cr}}

I_{cc} : 100% des tâches requérent une compétence commune.
I_{cr} : 95% des tâches requérent une compétences communes et 5% une compétence rare.
La probabilité qu'un employé maîtrise une compétence rare est de 20%, contre 80% pour une compétence commune.

TABLE 2.1 – Caractéristiques principales des instances.

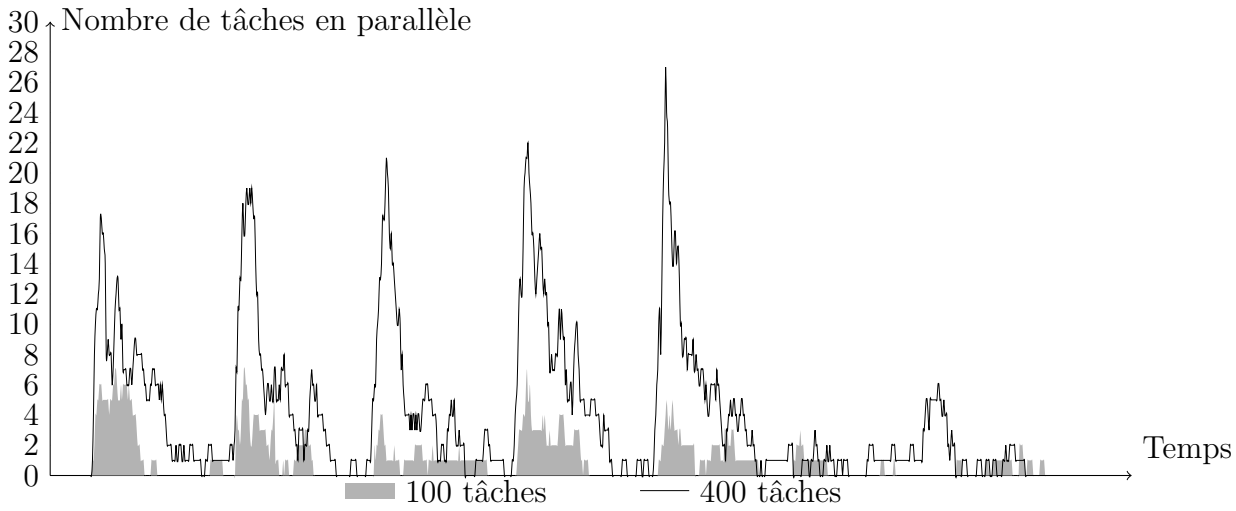


FIGURE 2.4 – Exemple d'une distribution de la charge de travail pour 100 et 400 tâches.

La génération des tâches est contrôlée via plusieurs paramètres qui permettent de jouer sur la fréquence d'apparition de tel ou tel type de tâche ainsi que sur leur positionnement dans le temps. La Figure 2.4 propose une illustration du profil de charge de travail obtenu, pour des instances à 100 et 400 tâches sur une semaine. Cette figure permet de mieux cerner les caractéristiques principales des instances : l'essentiel de l'activité se situe sur la semaine, avec des pics d'activités autour de 8h du matin. Notons également que d'un jour à l'autre, la charge de travail est relativement stable. Plus précisément, 90% des tâches sont générées en semaine (du lundi au vendredi). Les tâches de la semaine ont une probabilité de 50% d'être du matin, de 40% d'être de l'après-midi et de 10% d'être de nuit. Les tâches du matin ont une date de début générée selon une loi normale centrée à 8h et tronquée de manière à ce que toutes les dates de début soient comprises entre 6h et 10h. Cela permet de rendre compte du pic d'activité observé sur le terrain aux alentours de 8h. Les tâches de l'après-midi ont une date de début générée selon une loi uniforme allant de 10h à 20h. Enfin, les tâches de nuit ont une date de début générée selon une loi uniforme allant de 20h à 6h le lendemain. Les tâches du week-end ont une date de début générée selon une loi uniforme couvrant tout le week-end. Trois types de tâches sont considérées : les petites tâches (5%), les tâches moyennes (65%) et les grandes tâches (30%). La durée des petites tâches est générée selon une loi uniforme allant de 5 à 15 minutes. La durée des tâches moyenne est générée selon une loi normale centrée autour de 50 minutes et tronquée de manière à ce que les durées des tâches s'étalent de 30 à 70 minutes. La durée des grandes tâches est générée selon une loi uniforme allant de 2 à 5 heures. D'un point de vue industriel, les grandes tâches correspondent à des actes médicaux de longue durée (*e.g.* Électrocardiogramme sur 45 minutes) ainsi qu'à des analyses médicales (*e.g.* Pipetage avec centrifugation et décantation). Les petites et moyennes tâches correspondent quant à elles aux actes cliniques. Selon l'organisation de l'étude et le nombre de volontaires, ces tâches sont alors plus ou moins longues.

Le nombre d'employés considérés est calculé en fonction du volume de travail total et du paramètre I_v. La génération des employés est contrôlée via plusieurs paramètres qui permettent de jouer sur leur niveau de polyvalence, leur objectif de travail ou leurs disponibilités. Les congés du personnel sont générés de manière à former des séquences logiques. La probabilité qu'un employé soit en congé un jour donné est de 10% s'il travaillait la veille, et de 70% s'il était déjà en congé la veille. La probabilité qu'un employé soit en réunion un jour donné est de 5%. En cas de réunion, la date de début est générée selon une loi uniforme allant de 8h à 14h, la durée de la réunion est générée selon une loi uniforme allant de 1h à 4h. La charge de travail idéale des employés suit une loi de probabilité uniforme allant de 10h à 15h. Les valeurs générées sont ensuite modifiées de manière à prendre en compte les congés. En effet, si un employé est en congé sur une partie de la semaine, son objectif de

FIGURE 2.5 – Exemple d'une distribution des charges médicales idéales.

travail doit alors être diminué. Une fois que les objectifs de travail ont été adaptés aux disponibilités des employés, il reste à les adapter à la charge de travail globale de manière à ce que la somme des objectifs de travail corresponde à la somme des durées des tâches cliniques. Cela se fait en modifiant tous les objectifs de travail d'une valeur δ. Notons que cette normalisation des objectifs de travail ne modifie ni la difficulté de l'instance, ni la valeur de son optimum. Au final, nous obtenons des charges idéales allant de 3 à 20 heures, ce qui représente une variation importante (mais conforme à la réalité), comme cela est illustré à la Figure 2.5.

3

État de l'art

Le problème que nous avons présenté au Chapitre 2 correspond à un problème de planification de l'activité du personnel. Bien que ces problèmes soient étudiés depuis plus de 60 ans, l'intérêt de la communauté scientifique à leur égard reste très vif encore aujourd'hui. Cela s'explique notamment par un contexte socio-économique en perpétuelle mutation, conduisant du même coup à un besoin de renouvellement et d'adaptation des techniques de planification. Nous présentons tout d'abord une vue d'ensemble des problèmes de planification de personnel avant d'étudier plus spécifiquement les problèmes d'affectation de tâches fixées dans le temps. Nous nous concentrons ici sur des problèmes connexes, sachant que les différentes méthodes développées durant la thèse sont positionnées par rapport à l'état de l'art au début des chapitres correspondants. Nous concluons ce chapitre en soulignant les singularités du problème étudié, mais également ses points communs avec les problèmes classiques de la littérature.

Sommaire

3.1 Panorama de problèmes de planification de personnel

3.1.1 Périmètre, classifications et contextes applicatifs

Les problèmes de gestion du personnel ont longtemps porté sur des aspects tactiques et stratégiques tels que le dimensionnement d'effectif ou la politique salariale, comme cela est rappelé dans [12].

Au fur et à mesure de l'accroissement des capacités de production et de l'évolution des marchés, les entreprises passent d'une production standardisée et quantitative à une production diversifiée et qualitative. Dans un tel contexte, il devient essentiel de prendre en compte les spécificités des employés de manière à mettre en place des systèmes de production flexibles. Cela conduit alors à l'essor des problématiques de planification de l'activité du personnel [78]. Cette famille de problèmes vise à organiser l'activité des employés à un niveau plus ou moins fin de manière à ce que l'entreprise soit en capacité de répondre au mieux à un besoin plus ou moins bien connu. Au fil du temps, des avancées scientifiques et de l'évolution des pratiques managériales, les recherches se sont portées sur des problèmes de plus en plus complets avec des applications de plus en plus diverses. Par conséquent, la littérature portant sur le sujet est aujourd'hui très abondante. À titre d'indication, notons en particulier que l'état de l'art proposé par Ernst *et al.* ? [83] recense plus de 700 références dans le domaine de la planification de personnel. Face à cette multitude de travaux, plusieurs classifications ont été proposées.

Baker [10] propose l'une des premières classifications en distinguant trois grands problèmes de planification de l'activité du personnel : le *Day-Off Scheduling Problem*, le *Shift Scheduling Problem* et le *Tour Scheduling Problem*. Le Day-Off Scheduling Problem s'intéresse au positionnement des jours de repos des employés sur un horizon d'une ou plusieurs semaines [66, 191], alors que le Shift Scheduling Problem vise à déterminer les dates de début et de fin des vacations des employés [20, 108, 171]. Le Tour Scheduling Problem aborde quant à lui ces deux problèmes simultanément, permettant ainsi d'accéder à des plannings potentiellement meilleurs [6, 56, 120]. Deux autres classifications, dédiées exclusivement au *Tour Scheduling Problem*, répertorient les travaux par méthode de résolution. La première, proposée par Bechtold *et al.* [19], distingue les méthodes de résolution par programmation linéaire des méthodes de construction. La seconde, proposée par Alfares [6], affine cette première classification en retenant dix méthodes différentes : heuristiques, métaheuristiques, programmation linéaire, programmation linéaire en nombres entiers, approche multicritère, construction/amélioration, décomposition, implicites, génération/résolution et divers.

Contrairement aux classifications d'Alfares [6] et de Bechtold *et al.* [19], Ernst *et al.* [81] ainsi que Van den Bergh *et al.* [189] s'intéressent au problème générique de planification de personnel. Van den Bergh *et al.* [189] recensent ainsi plus de 300 travaux, répertoriés selon plusieurs axes tels que les caractéristiques des employés et des vacations, le type de décisions prises, la flexibilité des horaires de travail, le degré d'incertitude considéré ou encore le domaine d'application . Ernst *et al.* [81] proposent quant à eux une classification reposant sur l'idée qu'un problème donné correspond toujours à une combinaison particulière de six modules de bases : modélisation de la demande, positionnement des jours de repos, sélection des horaires de travail, construction de plannings individuels, affectation de tâches et affectation d'employés. Le module de modélisation de la demande consiste à traduire une estimation de l'activité de l'entreprise en tâches à réaliser ou bien directement en besoins en employés. Lorsque l'activité est relativement homogène, il est possible d'exprimer les besoins en personnel pour différents types de vacations fixées a priori (*e.g.* les centres hospitaliers). En revanche, lorsque l'activité de l'entreprise est très variable, il peut être nécessaire de spécifier les besoins en personnel pour un ensemble de créneaux horaires unitaires (*e.g.* les centres d'appels téléphoniques ou les restaurants). Parfois, il est même nécessaire d'exprimer la demande sous la forme de tâches bien précises se caractérisant par une fenêtre horaire d'exécution ou des dates de début et de fin fixées, une localisation géographique ainsi que des compétences requises. Une telle précision apparaît notamment dans les transports (*e.g.* les aéroports et les gares). Les modules de positionnement des jours de repos et de sélection des horaires de travail correspondent respectivement au Day-Off Scheduling Problem et au Shift Scheduling Problem. Le module de construction de plannings individuels consiste à élaborer un ensemble de séquences composées d'horaires de travail et de périodes de repos afin de couvrir correctement la demande. Chaque séquence correspond alors à un planning individuel,

mais n'est pas forcément attribuée à un employé précis. Cette dernière étape correspond au module d'affectation d'employés. Pour finir, le module d'affectation de tâches consiste à fixer les différentes activités réalisées par un employé donné au sein d'une vacation donnée.

Tous ces travaux illustrent bien la grande diversité des problèmes de planification de personnel qui s'explique principalement par l'évolution des pratiques managériales, la multitude des contextes applicatifs, ainsi que par les différences de législation selon les pays et les secteurs. Remarquons en particulier que la prise en compte de contexte industriel précis a souvent donné lieu à l'émergence de problèmes à part entière tels que la planification du personnel hospitalier avec le *Nurse Rostering Problem* [46] ou bien la planification d'équipages avec le *Crew Pairing Problem* [15, 57, 181, 190] et le *Crew Rostering Problem* [102, 121]. Ce dernier problème se décline d'ailleurs selon les différents modes de transport, avec notamment le *Bus Driver Scheduling Problem* pour le transport routier [142, 147, 195], le *Train Driver Scheduling Problem* pour le transport ferroviaire [172], *etc.* On parle également du *Ground-Crew Rostering Problem* [63, 65], par opposition au *Crew Rostering Problem*, afin de distinguer le personnel restant au sol, mais dont l'activité est tout de même rythmée par les arrivées et les départs des équipages. En raison de cette verticalisation, il existe de nombreuses correspondances entre les différents problèmes de planification de personnel [81]. Par exemple, le *Tour Scheduling Problem* est similaire au *Crew Rostering Problem*, à la différence près que le premier considère une activité variable alors que le second considère des séquences de trajets. De la même manière, le *Shift Scheduling Problem* est similaire au *Crew Pairing Problem*. Par ailleurs, certains travaux portant sur le *Tour Scheduling Problem*, planifient également les activités des employés en plus de leurs horaires de travail, ce qui s'apparente alors à un problème d'emploi du temps (*Employee Time Tabling Problem* [61, 71, 149, 177]).

Ce rapide panorama montre bien la difficulté d'établir une cartographie complète des problèmes de planification de personnel. Afin de positionner notre cas d'étude par rapport à la littérature, nous nous focaliserons donc sur les problèmes qui semblent les plus significatifs pour notre étude. Plus précisément, nous commençons par le *Nurse Rostering Problem*, dont le contexte métier rappelle celui de notre cas d'étude, puis nous abordons successivement le *Tour Scheduling Problem* et le *Crew Rostering Problem*. De cette manière, nous balayons les différentes modélisations possibles de la demande en personnel [81] (besoins exprimés par vacations, besoins flexibles, besoins définis par un ensemble de tâches).

3.1.2 Besoins exprimés par vacations, cas du *Nurse Rostering Problem*

Définition du Nurse Rostering Problem

Le Nurse Rostering Problem (NRP) est un problème de planification de personnel se focalisant sur l'environnement de travail bien particulier des établissements de santé tels que les hôpitaux ou les cliniques. Les horaires réalisables par les employés sont prédéfinis par un petit nombre de vacations possibles définissant complètement les journées de travail. Il est par exemple très fréquent de considérer trois vacations de huit heures correspondant à des horaires précis et couvrant l'ensemble d'une journée calendaire. Les plannings des employés forment alors des séquences de vacations et les besoins en personnel s'expriment en nombre d'employés par jour et par vacation. Remarquons qu'il est éventuellement possible de distinguer les besoins en personnel selon quelques catégories comme par exemple « infirmier-chef », « infirmier » et « infirmier-étudiant ». La singularité de cet environnement de travail provient des accords salariaux définissant des contraintes très particulières sur les plannings des employés. Ces contraintes varient selon les établissements, ce qui signifie qu'il est difficile d'énoncer un problème générique. La Table 3.1 propose un exemple de ce problème sur un horizon d'une semaine, avec trois vacations (Matin, Soir et Nuit). Le premier tableau donne tout d'abord les besoins en employés par jour et par vacation. Par exemple, il faut au minimum un employé

Vacation	Lundi	Mardi	Mercredi	Jeudi	Vendredi	Samedi	Dimanche
Matin (M)	2	1	0	1	0	1	1
Soir (S)	0	1	2	1	1	0	0
Nuit (N)	1	1	0	0	1	0	1

Données d'entrée : la demande en personnel, exprimée en vacations par jour

$(*, M, -, M, *)$	$(*, S, -, S, *)$	$(*, N, -, N, *)$
$(*, M, -, S, *)$	$(*, S, -, M, *)$	$(*, N, -, S, *)$
$(*, M, -, N, *)$	$(*, S, -, N, *)$	$(*, N, -, M, *)$
$(*, N, M, *)$	$(*, N, S, *)$	

Liste des séquences interdites ($-$: repos, $*$: une ou plusieurs vacations quelconques)

Personnel	Lundi	Mardi	Mercredi	Jeudi	Vendredi	Samedi	Dimanche
1	M	M	S	S	-	-	N
2	N	N	-	-	S	M	M
3	M	S	S	M	N	-	-

Solution : le planning du personnel exprimé en vacations par jour

TABLE 3.1 – Exemple d'un Nurse Rostering Problem

pour l'horaire de nuit du lundi. Le deuxième tableau donne les séquences de vacations interdites : il est interdit d'enchaîner une vacation Nuit avec une vacation Matin ou Soir et il est aussi interdit de positionner un jour de repos isolé. Le troisième tableau donne une solution au problème.

Quelques travaux portant sur le NRP

Nous nous concentrons ici sur les différentes manières dont les vacations et les compétences sont traitées dans le cadre du NRP, mais Burke *et al.* [46] ainsi que Cheang [59] proposent chacun un état de l'art complet sur le sujet. Remarquons tout d'abord que la difficulté principale du NRP est de gérer efficacement les nombreuses contraintes de séquence portant sur les plannings individuels. Autrement dit, le problème n'est pas tant de couvrir correctement la demande en personnel que de le faire de la meilleure manière possible du point de vue des employés.

Dans le cadre du NRP, le nombre de vacations considérées se réduit souvent à trois vacations travaillées [77, 139, 155, 156, 159, 186, 194]. Certains travaux considèrent cependant un plus grand nombre de vacations [43, 116, 118, 170], avec un maximum d'une quinzaine de vacations différentes. Dans l'ensemble, il est possible de conclure que la planification de l'activité de personnel hospitalier peut se faire avec un petit nombre de vacations différentes, ce qui permet de maîtriser une partie de la combinatoire du problème.

Puisque l'activité des employés au sein d'une vacation n'est pas explicitement planifiée, il est généralement inutile de prendre en compte les compétences des employés. Il faut en effet garder à l'esprit que les besoins en personnel médical sont estimés a priori et peuvent être en décalage par rapport aux besoins réels. Il est par conséquent logique de conserver une certaine flexibilité vis-à-vis des activités de travail des employés. Par ailleurs, les employés peuvent être suffisamment polyvalents pour qu'il soit inutile de distinguer le profil de demande selon chaque compétence. Pour ces raisons, de nombreux travaux ne prennent pas en compte la notion de compétences [24, 38, 44, 49, 155, 156, 186]. En revanche, d'autres travaux prennent en considération l'expérience des employés en distinguant par exemple les chefs de service, les employés très expérimentés, les employés peu expérimentés, ou

encore les employés en cours d'apprentissage [2, 3, 4, 28, 43, 45, 47, 77, 116, 118]. La demande est alors exprimée pour chaque niveau d'expertise.

Une particularité du NRP est de considérer un nombre important de contraintes souples, *i.e.* pouvant être violées mais entrainant alors une pénalité au niveau de l'objectif [35, 48, 60]. Ces contraintes servent typiquement à prendre en compte les souhaits des employés ainsi qu'à réduire la pénibilité des horaires de travail. Les souhaits des employés portent généralement sur leurs horaires de travail, *e.g.* tel employé souhaite éviter les horaires de nuit, alors que tel autre voudrait être en repos chaque weekend. La pénibilité du travail est une notion moins subjective qui est généralement définie pour tous les employés. Tel établissement souhaite par exemple que les employés ayant réalisé plusieurs vacations de nuit puissent bénéficier d'au moins deux jours de repos, alors que tel autre souhaite que chaque employé bénéficie d'au moins un weekend entièrement reposé tous les mois. D'un problème à l'autre, ces contraintes peuvent être souples ou dures, voire éventuellement contradictoires, ce qui souligne la diversité des problèmes regroupés au sein du NRP.

Différences et point communs avec le cas d'étude

Étant donné que notre problème porte également sur la planification du personnel en milieu médical, il semble important de le positionner par rapport au NRP. Les deux problèmes visent à planifier l'activité du personnel de manière à couvrir une certaine demande. De plus, les deux problèmes prennent en compte la qualité des plannings du point de vue des employés. Cependant, la manière dont ces deux objectifs sont abordés dans le NRP diffèrent complètement de notre cas d'étude. Dans le cas du NRP, la demande s'exprime en nombre d'employés par vacation et éventuellement par niveau d'expérience alors que dans notre cas d'étude, il faut affecter à chaque employé des tâches fixées dans le temps et requérant des compétences très particulières, ce qui nécessite un grand nombre de vacations *ad hoc*, comme nous le verrons au Chapitre 5. Par ailleurs, le NRP comprend également de nombreuses contraintes de séquences visant à produire des plannings individuels satisfaisant pour les employés. Dans notre cas d'étude, ces contraintes n'apparaissent pas. Cela s'explique notamment par l'absence d'horaires prédéfinis et donc de « patrons » à respecter. Cette grande flexibilité permet à Biotrial de mieux couvrir toutes les tâches cliniques, ce qui est un véritable challenge. La qualité des plannings est cependant prise en compte via une répartition équitable de la charge globale de travail (*cf.* Section 2.1.5).

3.1.3 Besoins flexibles, cas du *Tour Scheduling Problem*

Définition du Tour Scheduling Problem

Le *Tour Scheduling Problem* (TSP) est un problème de planification de personnel accordant une attention particulière à la construction des horaires de travail. Il s'agit d'une part de fixer les jours de repos des employés, mais également de déterminer les horaires de travail réalisés. Ce problème apparaît dans de très nombreux contextes et présente une multitude de variantes. Par exemple, selon les travaux, les vacations peuvent être construites durant la résolution du problème, sélectionnées parmi un ensemble de vacations possibles ou bien données en entrée du problème. En plus des horaires de travail, certaines variantes du TSP spécifient également les activités des employés. Lorsque les employés ont des compétences et des disponibilités spécifiques et lorsque les vacations sont construites finement, on obtient alors une variante du TSP proche de notre cas d'étude. La Table 3.2 propose un exemple de TSP considérant deux compétences distinctes, A et B, ainsi qu'une demande en personnel sur un horizon d'une journée, avec des créneaux d'une heure. Ainsi, il faut par exemple un employé maîtrisant la compétence A et deux employés maîtrisant la compétence B pour le créneau horaire de 8h. Les vacations ont une amplitude maximale de huit heures et présentent une pause d'une heure entre 12h et 14h. Le deuxième tableau donne les disponibilités horaire de chaque employé ainsi que

Activité	7h	8h	9h	10h	11h	12h	13h	14h	15h	16h	17h
A	1	1	2	2	2	2	1	1	1	0	0
B	0	2	2	3	3	2	2	1	1	1	1

Données d'entrée : besoin en employés par activité et par créneau horaire

Employé	7h	8h	9h	10h	11h	12h	13h	14h	15h	16h	17h
e_1 (A,B)	0	0	0	1	1	1	1	1	1	1	1
e_2 (B)	1	1	1	1	1	1	1	1	1	1	1
e_3 (A)	1	1	1	1	1	1	0	0	0	0	0
e_4 (A,B)	1	1	1	1	1	1	1	1	1	1	1
e_5 (A,B)	0	0	1	1	1	1	1	1	0	0	0

Contraintes de compétence et de disponibilité

Employé	7h	8h	9h	10h	11h	12h	13h	14h	15h	16h	17h
e_1				B	B	B	B		B	B	B
e_2		B	B	B	B	B		B			
e_3	A	A	A	A	A	A					
e_4		B	A	A	A		B	A	A		
e_5			B	B	B	A	A				

Solution : activités réalisées par les employés pour chaque créneau horaire

TABLE 3.2 – Exemple d'un *Tour Scheduling Problem*

ses compétences. Par exemple, l'employé e_1 maîtrise les compétences A et B et n'est pas jusqu'à 10h. Le troisième tableau donne une solution au problème.

Quelques travaux portant sur le TSP

Comme cela est souligné par Mabert et Watts [145], un plus grand degré de liberté dans la construction des horaires du personnel permet une gestion plus fine de l'effectif, ce qui peut conduire à un accroissement de la productivité, mais également entrainer un accroissement de la pénibilité du travail. En effet, le fait de gérer finement l'effectif de travail permet de couvrir au mieux l'activité, ce qui réduit d'autant les périodes d'inactivité des employés. Cela conduit également à construire des horaires de travail très variables et à densifier les journées de travail des employés. De manière à profiter de l'accroissement potentiel de productivité, il faut donc veiller à réduire également la pénibilité des conditions de travail des employés, ce qui requiert d'établir un bon compromis entre ces deux intérêts. Les auteurs fondent leur réflexion sur une étude comparative de différentes procédures permettant d'explorer la frontière du bon compromis entre les intérêts de l'entreprise et ceux des employés. La gestion fine des employés est particulièrement pertinente lorsque l'activité considérée est très variable, ce qui est le cas pour de nombreuses sociétés de services par exemple. Ainsi, au fil du temps, de nombreux travaux ont considéré des vacations de plus en plus flexibles, mais également des effectifs de plus en plus hétérogènes permettant ainsi de mieux prendre en compte les besoins des entreprises ainsi que les spécificités des individus. Nous proposons ici d'étudier quelques-unes des variantes du TSP selon trois axes principaux, portant respectivement sur l'hétérogénéité de l'effectif de travail, le détail de construction des vacations et les contraintes prises en compte lors de la construction des vacations et du planning. Une analyse plus fine des différents travaux portant sur le TSP est donnée par Alfares [6] qui propose un état de l'art des méthodes de résolution développées pour le TSP ainsi qu'un ensemble de tableaux comparatifs permettant d'identifier les différentes variantes étudiées.

En ce qui concerne l'hétérogénéité des employés, remarquons tout d'abord que de nombreux travaux portant sur le TSP se sont concentrés sur le cas d'un effectif homogène [5, 11, 16, 39, 58, 89, 113, 134]. Un premier niveau de complexité supplémentaire consiste à distinguer différents groupes d'employés partageant certaines caractéristiques importantes. Il est par exemple possible de faire la distinction entre les employés à temps plein et ceux à temps partiel. Les premiers réalisent alors des vacations de 8 ou 9 heures avec, la plupart du temps, une pause déjeuner alors que les seconds réalisent des demi-vacations de 4 ou 5 heures [40, 42, 56, 173]. Afin de gagner en flexibilité, il est également possible de considérer que les employés à temps partiel réalisent des vacations de durée variable [115]. Il est également possible de regrouper les employés selon leur statut ou leur degré d'ancienneté [98]. Cette distinction peut servir notamment à départager les employés capables de réaliser les tâches compliquées de ceux qui sont encore en formation. En poursuivant cette logique de différenciation, il est possible de considérer chaque employé comme un individu à part entière. La plupart des travaux atteignant ce niveau de détail différencient les employés via leurs compétences [18, 50, 120, 141, 143, 153], leurs disponibilités [8, 120, 141, 143, 153] ou leurs préférences [120, 153].

Lorsqu'il s'agit de construire des vacations, il est tout d'abord possible de s'intéresser au profil des vacations, *i.e.* leurs dates de début, leurs durées et le positionnement des pauses. Par exemple, Jarrah *et al.* [115] considère des vacations de durée variable avec une pause déjeuner flottante. D'un point de vue pratique, il peut être plus efficace de réduire le niveau de flexibilité de création des vacations, ce qui peut d'ailleurs coïncider avec un ensemble de pratiques managériales. Par exemple, Rong *et al.* [174] considère un TSP avec un seul type de vacation par jour, mais avec une pause déjeuner flottante. Gopalakrishnan *et al.* [103] considèrent des vacations avec des dates de début fixées, mais une durée variable. A l'inverse, Brusco et Johns [42] considèrent des vacations avec une durée fixe, mais une date de début variable. En plus d'une gestion fine du profil des vacations, un second niveau de détail consiste à distinguer également les différentes activités réalisées au sein de chaque vacation. Certains travaux considèrent que la même activité est réalisée durant toute la durée d'une vacation [120, 143, 173] alors que d'autres autorisent les changements d'activités [84, 110, 141]. Par ailleurs, Brucker et Qu [37] étudient un problème d'affectation de tâches fixées, non préemptables et requérant des compétences spécifiques tout en construisant les vacations du personnel sur une journée.

Lorsqu'il s'agit de construire un planning individuel, plusieurs contraintes peuvent rentrer en ligne de compte. En effet, lorsque les vacations des employés ne sont pas prédéfinies, il n'est pas possible de définir les séquences de vacations interdites, comme cela est le cas pour le NRP. Afin de construire des plannings valides, il est alors nécessaire de contraindre par exemple l'amplitude des vacations et leurs dates de début, mais également le positionnement des pauses ainsi que le nombre maximum de vacations consécutives travaillées. Il est par exemple très fréquent d'imposer un nombre minimum de jours de repos (potentiellement consécutifs) par semaine ou par mois [173, 174]. Il est également possible de contraindre les dates de début des vacations de différentes manières. À titre d'exemple, Brusco *et al.* [42] limitent le nombre de dates de début différentes de manière à éviter des arrivées et des départs continuels tout au long de la journée, alors que Cezik *et al.* [56] cherchent à obtenir des plannings individuels pour lesquels les dates de début des vacations sont proches. De manière similaire, il est fréquent d'imposer une durée minimale aux vacations des employés [141, 143]. Selon le degré de flexibilité de la date de début des vacations, il peut être nécessaire ou non de vérifier le repos quotidien entre deux vacations travaillées. Par exemple, si les vacations ne commencent qu'entre 7h et 12h et que leur amplitude est limitée à 8h, leur fin au plus tard est alors de 20h, laissant ainsi un repos minimum entre deux journées de travail de 11h. Dans une telle configuration, il suffit donc de vérifier que chaque employé ne travaille pas plus d'une vacation par journée de travail pour garantir en même temps le repos quotidien. En revanche, lorsque les vacations peuvent commencer à toute heure du jour ou de la nuit, il faut alors garantir un repos minimum pour chaque employé. Un premier

moyen de garantir ce repos consiste simplement à poser une contrainte de distance entre la fin de la vacation k et le début de la vacation $k+1$. Une deuxième méthode consiste à imposer un minimum de régularité au niveau des horaires de chaque employé. Il est par exemple possible de contraindre les employés à commencer toutes leurs vacations d'une semaine à la même heure, ce qui permet de respecter automatiquement le repos quotidien.

Un cas d'application intéressant du TSP apparaît au sein des aéroports, lorsqu'il s'agit de planifier l'activité du personnel non navigant (*Ground Crew Rostering Problem*), tel que les bagagistes, les techniciens et ingénieurs en charge de la maintenance des avions ou le personnel chargé des opérations de contrôle et d'embarquement [63, 72, 73, 111, 148, 158, 178]. Nous identifions ce type de problèmes comme une application du TSP car le travail des employés au sol est rythmé par les départs et arrivées des avions, entraînant une activité très variable d'un jour à l'autre et d'une heure à l'autre, dont la couverture fine requiert une grande flexibilité lors de la construction des vacations. Un vaste état de l'art, ainsi que plusieurs méthodes de résolution portant sur ce problème sont proposés dans la thèse d'Herbers [109]. Puisqu'il s'agit d'un problème pratique, de nombreux travaux présentent des outils d'aide à la décision implémentés pour un contexte spécifique [63, 72, 73, 111, 178]. Selon les travaux, le problème étudié est par conséquent très variable. Par exemple, Dijkstra *et al.* [72] considèrent un problème dans lequel l'effectif de travail est organisé en une douzaine d'équipes, chaque employé d'une même équipe travaillant aux mêmes horaires, alors que Brusco *et al.* [41] construisent des plannings individuels, en imposant à un même employé de commencer toutes ses vacations à la même heure. Brusco *et al.* [41] ainsi que Schindler *et al.* [178] considèrent un nombre variable de vacations avec une demande exprimée par créneaux horaires de 15 minutes, alors que Dijkstra *et al.* [72] et Nobert *et al.* [158] considèrent des vacations fixées mais une demande très fine exprimée comme une quantité de travail à réaliser durant une fenêtre horaire. Autrement dit, la date de début de la tâche n'est pas complètement fixée, et sa durée varie en fonction du nombre d'employés qui y sont affectés. Remarquons que ces travaux sont à la limite de ce qu'on peut considérer comme un problème avec demande flexible, car ils introduisent la notion de tâche. Cela s'explique par le contexte applicatif, qui permet d'estimer avec une grande précision la charge à couvrir. De manière analogue, Dowling *et al.* [76] proposent un outil d'aide à la décision composé de deux modules : le premier module construit le planning général des employés avec une précision d'une demi-heure sur un horizon d'un mois à partir d'une estimation des besoins en personnel sans prendre en compte les compétences ; le second module permet d'affecter un ensemble de tâches fixées dans le temps aux employés compétents sur un horizon d'une journée, avec horaires de travail fixés.

Différences et point communs avec le cas d'étude

Le TSP est à notre connaissance le problème de planification de personnel accordant le plus haut degré de flexibilité et de détail vis-à-vis de la construction des vacations du personnel, ce qui se rapproche de notre propre cas d'étude. Par exemple, certains auteurs planifient l'activité du personnel sur des créneaux de 15 à 30 minutes, sur un horizon d'une semaine, en considérant les compétences et les disponibilités des employés [141, 143]. Certaines approches définissent ainsi les horaires de travail ainsi que les activités réalisées par chaque employé durant chaque créneau horaire, ce qui s'apparente à notre propre configuration, dans la mesure où cela permet à un employé de réaliser différentes activités sur des intervalles horaires fixés. D'autres travaux prennent en compte une demande encore plus fine qui se rapproche de notre notion de tâche [37, 72, 76, 158]. Malgré ces points communs, notre cas d'étude diffère du TSP sur au moins un point essentiel : l'activité des employés. Dans le TSP, la demande est exprimée sous la forme d'un nombre d'employés par type d'activité et par créneau horaire ce qui conduit à considérer soit des tâches unitaires commençant à des intervalles réguliers, soit des tâches préemptables. Dans le cadre de la planification du personnel de Biotrial, nous sommes amenés à considérer des tâches cliniques avec des durées très différentes, pouvant commencer à tout

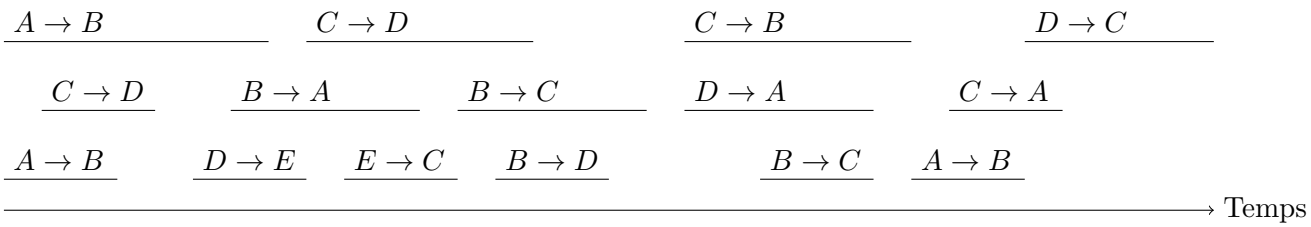
moment de la semaine et devant être réalisées sans préemption. Par conséquent, le cadre général du TSP n'est pas approprié pour notre configuration particulière.

3.1.4 Besoins sous forme de tâches, cas du *Crew Scheduling Problem*

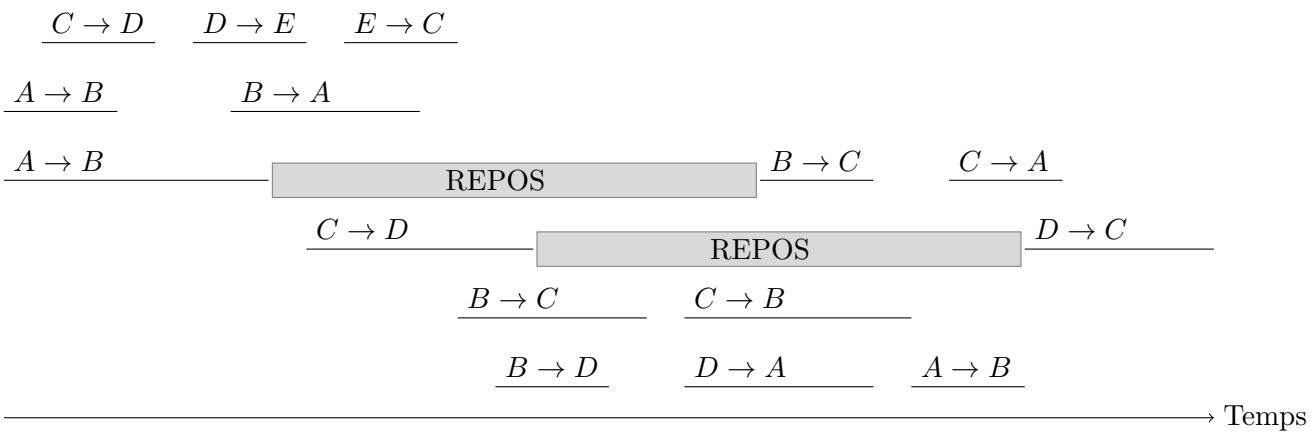
Définition du Crew Scheduling Problem

Les problèmes de planification d'équipage, ou *Crew Scheduling Problem* (CSP), abordent la planification du personnel navigant, dont les plannings sont constitués d'une succession de trajets fixés dans le temps. Ce problème est généralement abordé en deux temps, au travers du *Crew Pairing Problem*, noté CPP et du *Crew Rostering Problem*, noté CRP.

Le CPP consiste à partitionner l'ensemble des trajets de manière à ce que chaque ensemble forme une rotation, *i.e.* une séquence de trajets formant un circuit géographiquement et temporellement cohérent. Une rotation correspond alors à un ou plusieurs jours de travail séparés par une période de repos, chaque jour de travail comprenant lui même un ou plusieurs trajets réalisables par un seul employé. Le point de départ d'une rotation est appelé une base. L'objectif du CPP est de créer un ensemble de rotations de coût minimal. La Figure 3.1 propose une illustration pour le CPP : l'ensemble des trajets considérés est donné à la Figure 3.1a alors que la Figure 3.1b propose une solution composée de 6 rotations, deux d'entres elles s'étalant sur deux jours et incluant donc un repos. Ainsi, par exemple, les trajets $C \Rightarrow D$, $D \Rightarrow E$ et $E \Rightarrow C$ sont regroupés en une seule rotation.



(a) Données d'entrée sur les trajets : intervalle, départ et arrivé.

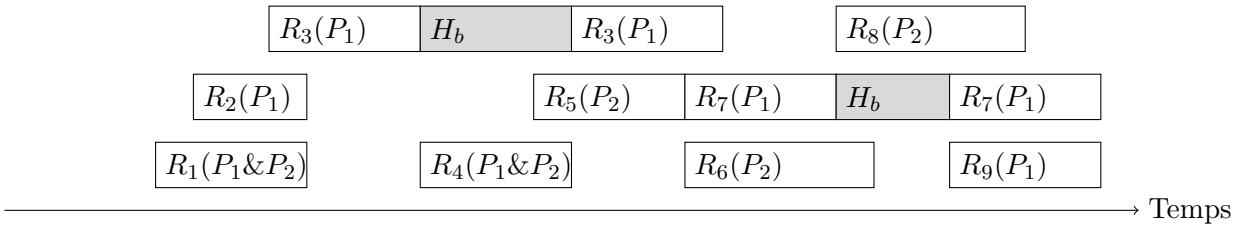


(b) Solution réalisable composée de 6 rotations (une rotation par ligne).

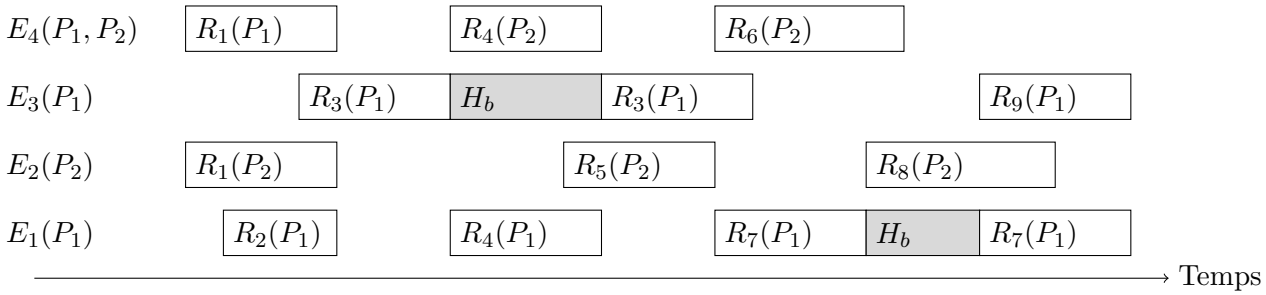
FIGURE 3.1 – Exemple de *Crew Pairing Problem*.

Le CRP consiste à construire des plannings individuels de manière à couvrir l'ensemble des rotations tout en respectant les contraintes légales et organisationnelles. Notons que chaque rotation peut nécessiter plusieurs employés à des postes différents comme par exemple dans les transports

aériens, où il faut un pilote, un copilote, ainsi qu'une équipe de stewards. Il est fréquent de distinguer deux variantes importantes du *CRP*. Une première variante, utilisée principalement en Amérique du Nord, consiste à construire les plannings individuels de manière anonyme et à laisser les employés se répartir eux-mêmes les plannings selon une procédure itérative souvent fondée sur l'ancienneté des employés. Cette méthode permet aux employés de choisir leur planning en toute transparence, mais cela peut entraîner un surcoût pour l'entreprise qui doit mobiliser davantage de personnel de manière à pallier les absences ou le manque de compétences des employés qui se retrouvent avec des plannings ne leur correspondant pas parfaitement. La seconde variante, utilisée en Europe, consiste à construire les plannings individuels et à les affecter directement aux employés en prenant en compte des données individuelles telles que les préférences, les disponibilités, et les compétences des employés. Cette méthode permet une gestion automatique de l'affectation des plannings aux employés, de manière à répartir équitablement les rotations les plus pénibles et la charge de travail en fonction des disponibilités et des compétences des employés, mais également en fonction de leurs souhaits. L'avantage de cette méthode est de réduire le coût de l'effectif en élaborant des plannings prenant en compte toutes les particularités du personnel. En revanche, une telle méthode peut conduire à une chute de la satisfaction du personnel dont les préférences sont souvent difficiles à gérer convenablement. La Figure 3.2 propose une illustration pour le *CRP* : les rotations considérées sont données à la Figure 3.2a alors que la Figure 3.2b propose une solution réalisable. Par exemple, l'employé E_4, pouvant occuper les postes P_1 et P_2, est affecté aux rotations R_1, R_4 et R_6, aux postes P_1, P_2 et P_2. Puisque les décideurs de Biotrial réalisent des plannings individuels pour chaque employé, nous nous concentrons ici sur la variante européenne du CRP.



(a) Données d'entrée sur les rotations : début, fin, poste ($P_1\&P_2$), repos hors base (H_b).



(b) Solution réalisable.

FIGURE 3.2 – Exemple de *Crew Rostering Problem*.

Quelques travaux portant sur le Crew Pairing/Rostering Problem

En ce qui concerne le CPP, les contraintes intervenant lors de la création des rotations sont très nombreuses et varient en fonction des cas d'étude. Plusieurs contraintes récurrentes peuvent cependant être mentionnées, comme par exemple le fait de limiter le temps cumulé des trajets par journée de travail et par rotation, le fait de respecter l'amplitude maximale des journées de travail et des rotations, mais également le repos minimal entre deux journées de travail ou encore le fait de respecter le temps de battement minimum entre deux trajets consécutifs et de réserver des périodes de travail

pour les réunions en début et fin de journée [57, 190]. En ce qui concerne l'objectif du problème, la majorité des travaux cherchent à minimiser le coût des rotations. Ce coût s'obtient généralement en considérant trois éléments différents : l'amplitude de la rotation, le temps cumulé de trajet de la rotation et le coût minimum par rotation des employés [15]. Il s'agit par conséquent de favoriser les rotations longues et comprenant un minimum de temps de battement. Un autre objectif intéressant consiste à produire des rotations robustes, en créant par exemple des rotations susceptibles d'être affectées à de nombreux employés [181]. Dans le cas particulier des vols long-courriers, il peut être intéressant de faire voyager le personnel en tant que passager de manière à le repositionner sur un aéroport à forte activité [14].

En ce qui concerne le CRP, notons tout d'abord que Kohl et Karisch [121] proposent une vue d'ensemble des grandes variantes du CRP et de ses méthodes de résolution dans le contexte aérien. Les auteurs listent notamment les principales contraintes légales et organisationnelles du CRP. Un premier ensemble de contraintes porte sur les trajets : les types d'employés requis, ainsi que leurs qualifications, la complémentarité globale de l'équipage et le nombre maximum d'employés inexpérimentés. Un deuxième groupe de contraintes concerne les employés : prise en compte des absences et des pré-affectations, mais également des incompatibilité inter-employés ou au contraire des associations obligatoires d'employés. Un dernier type de contraintes prend en compte la pénibilité du travail : repos minimum entre deux rotations consécutives en fonction du type des vacations, nombre minimum de jours de repos, temps de vol maximal, nombre maximal de changements de fuseaux horaires, nombre maximum d'horaires de nuit, *etc.* Parmi toutes ces contraintes, les plus fréquemment étudiées portent sur les compétences des employés [69, 119, 135, 144, 154], leurs pré-affectations et indisponibilités [119, 144, 154] et leurs préférences [144, 154]. En ce qui concerne la qualité du planning, Kohl et Karisch [121] distinguent quatre types d'objectif pertinents. Un premier objectif porte sur la minimisation du coût de couverture des rotations (heures supplémentaires, embauches, annulation des vols). Un deuxième objectif porte sur la maximisation de la robustesse du planning et un troisième sur la répartition équitable de la charge de travail et de sa pénibilité. Le dernier objectif prend en compte le respect des préférences individuelles. Freling *et al.* [96] identifient des objectifs similaires mais regroupent la répartition équitable du travail et le respect des préférences individuelles au sein d'un unique objectif de bien-être des employés. De manière générale, le bien être des employés est l'un des éléments importants du CRP. En ce qui concerne le respect des préférences individuelles par exemple, Dawid *et al.* [69] prennent en compte les préférences des employés et proposent une technique permettant de mieux équilibrer la charge de travail en fonction des différents grades des employés tout en facilitant l'obtention d'une solution réalisable. En ce qui concerne la répartition équitable du travail, remarquons que l'utilisation de plannings cycliques constitue une solution idéale [51, 53], mais limitée au cas où l'activité de la compagnie de transport est elle-même cyclique. L'objectif est alors de minimiser le nombre d'employés nécessaires. Dans le cas d'une activité acyclique, l'équité peut être abordée via la minimisation des déviations individuelles de tel ou tel critère par rapport aux standards moyens [146].

Certains travaux abordent à la fois le CPP et le CRP. Par exemple, Chabrier [57] propose une approche en deux phases reprenant la séparation classique CPP/CRP. À l'inverse, Freling *et al.* [96] proposent une méthode de résolution pour le problème de planification d'équipages pris dans son ensemble (CPP & CRP). Les auteurs comparent cette méthode à une décomposition classique et observent une amélioration des résultats par rapport à l'approche séquentielle au prix d'un temps de calcul plus important.

Différences et points communs avec notre cas d'étude

Les problèmes de planification d'équipages sont très proches de notre cas d'étude car ils abordent à la fois le problème de création des plannings individuels, mais également le problème de couverture de tâches fixées dans le temps. Certains travaux abordent ces aspects simultanément, ce qui correspond au mode de planification de Biotrial. Contrairement au NRP et au TSP, les vacations des employés correspondent à des tâches précises ne pouvant être préemptées et chaque employé ne peut réaliser qu'une seule tâche à la fois.

Une première différence importante porte sur la nature des tâches : dans le cas du CSP, les tâches correspondent à des trajets et il est par conséquent nécessaire de vérifier la cohérence géographique des séquences de trajets affectées à un même employé, ce qui n'est pas le cas dans notre problème. De plus, de nombreuses contraintes découlant de cette donnée géographique, telles que le nombre maximum de changements de fuseau horaire, n'ont pas de sens dans le cadre de notre étude.

Une seconde différence porte sur l'évaluation du coût d'une rotation. Comme nous avons pu le voir, une rotation est d'autant plus efficiente qu'elle comprend peu de temps de battement, ce qui ne correspond pas à notre cas d'étude. Dans le cas de la planification du personnel de Biotrial, il est au contraire important de garantir du temps de travail non affecté, de manière à ce que les employés puissent vaquer à leurs activités administratives. Par ailleurs notre fonction objectif ne prend pas en compte le coût individuel d'une vacation, mais évalue à la place le coût de l'ensemble des vacations, car nous mesurons l'équité non pas par rapport à des standards moyens bien établis, mais par rapport au respect d'une charge cible individuelle variant selon l'activité de l'entreprise. Même si on se limite au problème de satisfaction, la structure des solutions varie fortement entre notre cas d'étude et le CSP. En effet, les plannings que nous construisons ne portent que sur une partie de l'activité des employés, conduisant ainsi à des plannings plus flexibles que ceux du CSP où l'activité de chaque ressource est complètement planifiée.

3.1.5 Gestion de l'équité en planification de personnel

Les problèmes de planification de personnel que nous avons présentés aux Sections 3.1.2 à 3.1.4 considèrent des contextes métiers très différents, ce qui signifie que les activités des employés et les contraintes qui régissent leurs plannings varient énormément. Ces problèmes ont cependant en commun un même objectif : organiser l'activité des employés de manière à obtenir des plannings économiquement efficaces et socialement appréciés. À ce titre, le partage équitable de la charge de travail apparaît comme une problématique centrale, dont nous dressons ici un rapide panorama.

Le principe d'équité apparaît naturellement dans le développement de nos sociétés car il permet d'évaluer le ressenti collectif d'un ensemble d'agents vis-à-vis du partage de ressources finies. Autrement dit, l'équité est un indicateur incontournable lorsqu'il s'agit d'évaluer le partage de ressources limitées. Le partage équitable des ressources peut se définir de la manière suivante :

« Étant donnés un ensemble d'objets et un ensemble d'agents ayant des préférences sur les parts possibles qu'ils peuvent recevoir, comment partager les objets entre les agents, de manière à ce que le partage soit le plus équitable possible ? » [31]

Ce problème est largement pluridisciplinaire puisqu'il relève à la fois de la philosophie et de la sociologie, mais également de l'économie et des mathématiques. Pour une étude plus approfondie de ces différents aspects, nous renvoyons à la thèse de Bouveret [31]. Nous nous concentrerons ici sur la gestion pratique de l'équité dans le contexte des problèmes de planification de personnel. Autrement dit,

nous nous intéressons ici aux différentes méthodes proposées pour construire des plannings équitables.

Une première manière d'aborder le problème de partage équitable des ressources consiste à construire des plannings cycliques [9, 16, 17]. Dans ce cas, chaque employé réalise à tour de rôle les différents horaires du cycle, si bien qu'au bout du cycle chaque employé a réalisé la même quantité de travail, aux mêmes horaires, et avec les mêmes séquences d'horaires. Remarquons tout d'abord que l'utilisation de plannings cycliques requiert une activité elle-même cyclique, ce qui limite l'intérêt pratique de cette méthode. Par ailleurs, l'attribution de plannings identiques ne conduit pas automatiquement à un état équitable car les individus ne sont pas forcément égaux initialement, comme cela est énoncé par Aristote :

« Les contestations et les plaintes naissent quand, étant égales, les personnes possèdent ou se voient attribuer des parts non égales, ou quand, les personnes n'étant pas égales, leurs parts sont égales. [...] Tous les hommes reconnaissent, en effet, que la justice dans la distribution doit se baser sur un mérite de quelque sorte, bien que tous ne désignent pas le même mérite. »
Aristote, Éthique à Nicomaque, Livre V, chapitre 6, traduction Tricot.

Autrement dit, la construction de plannings équitables est particulièrement problématique lorsque les employés sont différents les uns des autres et/ou lorsqu'il n'est pas possible de recourir à des plannings cycliques. Dans un tel contexte, chaque employé a sa propre perception de la qualité générale du planning, et il devient alors très difficile pour un décideur unique d'aboutir à un planning équitable. De manière à prendre en compte le ressenti des employés, il est tout d'abord possible de les solliciter durant la phase de construction du planning général, au moyen d'un système d'enchères : chaque employé dispose d'un capital de points et peut faire des offres pour telle ou telle vacation, jusqu'à ce que toutes les vacations soient attribuées [102, 121]. Remarquons tout d'abord, qu'il faut au préalable accorder un capital de points à chaque individu, ce qui nous ramène au problème initial. De plus, ce genre de système entraîne souvent un manque d'efficacité puisque chaque individu poursuit alors ses propres objectifs, sans avoir conscience des enjeux globaux [121].

Pour obtenir un planning à la fois très efficace économiquement et très apprécié socialement, de nombreuses approches construisent directement le planning général, en différenciant les employés selon certains critères, jugés pertinents par rapport au contexte de travail. Une première méthode consiste à distinguer les employés selon leur contrat de travail, leur grade ou leur ancienneté, ce qui fait alors émerger des classes d'employés avec des besoins spécifiques [84, 154]. Cependant, au sein de chaque classe d'employés, la problématique d'équité reste entière. Pour corriger cela, un second niveau de détail consiste à distinguer chaque employé via ses préférences. Dans la plupart des travaux, les préférences portent sur les horaires de travail [83]. Une préférence peut concerner un jour précis, par exemple, tel employé souhaite être en repos le 2 juillet, ou bien porter sur une période plus ou moins longue, par exemple, tel employé voudrait travailler de nuit jusqu'à la fin du mois alors que tel autre voudrait éviter les horaires d'après-midi chaque mercredi. Lorsque le décideur dispose des préférences de chaque individu, il cherche alors à déterminer un indicateur de qualité indexé sur le degré de prise en compte des préférences.

Pour évaluer la prise en compte des préférences, une manière classique de procéder consiste à attribuer une pénalité variable pour chaque préférence non respectée. La pénalité est alors d'autant plus forte que le non respect de la préférence correspondante est dommageable. Plusieurs éléments peuvent intervenir dans le calcul des pénalités, comme par exemple la nature elle même de la préférence, ou bien le grade ou l'ancienneté de l'employé y correspondant. L'objectif est alors de minimiser la somme des pénalités [35, 48, 60]. Cette méthode revient cependant à considérer qu'une amélioration minime du planning individuel avec la plus grande pénalité doit être rejetée dès lors qu'elle

conduit à une dégradation plus importante de plannings moins pénalisés. Autrement dit, cette approche permet d'obtenir des plannings très satisfaisants pour la plupart des employés, mais n'offre aucune garantie d'équité.

Une autre méthode consiste à indexer la qualité générale du planning global sur la qualité du planning individuel le moins apprécié [160]. Cela se traduit par une fonction objectif de type *min max*. De cette manière, une amélioration minime du planning le plus pénalisé est acceptée même si elle entraîne des pénalités importantes sur d'autres plannings, initialement de meilleure qualité. Autrement dit, cette approche permet de réduire les écarts de qualité entre les plannings individuels, mais cela peut conduire à une détérioration générale de la qualité du planning.

Le critère d'équité que nous avons retenu à la Section 2.2.3 s'apparente aux méthodes indexant la qualité générale du planning à la qualité des plannings individuels les moins bons. Dans notre cas, il s'agit des plannings correspondant aux employés les plus éloignés de leur charge idéale. Le fait de se focaliser sur le pire cas ne conduit pas à une dégradation générale du planning puisque ce qui est gagné par un employé est exactement perdu par un autre. En effet, lorsqu'on déplace une tâche d'un employé travaillant trop à un employé ne travaillant pas assez, les deux employés se rapprochent alors de leurs objectifs de travail de manière égale. En cas de surcharge généralisée, le fait de déplacer une tâche d'un employé en très grande surcharge de travail à un employé en grande surcharge de travail, revient alors à répartir plus équitablement un effort ponctuel, et réciproquement, en cas de sous-charge générale de travail, cela revient à répartir équitablement l'allègement de la charge de travail.

3.2 Panorama de problèmes d'affectation de tâches fixées

Mis à part le cas particulier des problèmes de planification d'équipages, nous avons pu voir à la Section 3.1 qu'il était assez rare d'attribuer aux employés des tâches fixées dans le temps et non préemptables, car cela requiert une bonne connaissance a priori des besoins en employés. En règle générale, l'activité des employés est donc organisée de manière plus grossière, ce qui permet d'ailleurs de gagner en flexibilité et en réactivité face aux aléas opérationnels. Puisque la notion de tâches fixées dans le temps et non préemptables est une donnée centrale de notre cas d'étude, nous nous focalisons à présent sur plusieurs problèmes d'affectation de tâches fixées. Contrairement à la section précédente, nous ne nous restreignons pas aux ressources humaines.

Le problème consistant à affecter un ensemble de tâches fixées dans le temps à un ensemble de ressources apparaît dans de nombreux contextes : dans les aéroports par exemple, les équipages doivent être affectés à des vols [53] ; dans les établissements scolaires, les cours sont attribués à des salles de classe [55] ; dans certaines usines, les tâches sont fixées dans le temps et doivent être affectées à des machines permettant de réaliser ces tâches [90] ; dans un contexte plus théorique, cela s'apparente à un problème de coloration de graphe d'intervalles (deux nœuds adjacents représentant deux tâches concomitantes doivent avoir des couleurs différentes) [100, 104]. Ces problèmes d'affectation prennent fréquemment en compte des restrictions vis-à-vis des possibilités d'affectation : l'équipage embarquant pour le vol à destination de Madrid doit parler espagnol ; le cours de chimie doit être affecté à une salle de classe avec des éviers ; pour plier telle ou telle pièce, il faut utiliser une presse d'une tonne ; chaque nœud ne peut être coloré qu'avec certaines couleurs. Nous nous intéressons par la suite à quelques problèmes de la littérature abordant ce sujet : le *Fixed Job Scheduling Problem* (FJSP), l'*Interval Scheduling Problem* (ISP) et le *Personnel Task Scheduling Problem* (PTSP).

Le FJSP est un problème de gestion de la production consistant à affecter un ensemble de tâches fixées dans le temps à un ensemble de ressources de manière à ce que deux tâches concomitantes ne

soient pas affectées à la même ressource. Selon le contexte, plusieurs objectifs sont envisageables, mais de manière générale, il s'agit soit de minimiser le coût des ressources permettant de réaliser toutes les tâches, soit de maximiser le profit réalisable par un ensemble de ressources. De nombreux cas particuliers sont bien sûr envisageables. Il est par exemple possible de distinguer les ressources selon de nombreuses caractéristiques telles que leur coût, leurs disponibilités horaires ou leurs compétences. Plus généralement, les données relatives aux disponibilités horaires et aux compétences peuvent être agrégées pour définir les ensembles de tâches réalisables par chaque ressource. Remarquons que deux états de l'art récents présentent un panorama des problèmes connexes au FJSP [125, 126], tels que le *K-Track Assignment Problem* [13], le *License Class Design Problem* [123], le *Class Scheduling Problem* [122] ou encore le *Class Room Assignment Problem* [55].

Dans un état de l'art relativement récent, Kolen *et al.* [125] définissent la version basique de l'ISP comme le problème consistant à trouver une affectation pour un ensemble de tâches non préemptables et fixées dans le temps à un nombre minimal de ressources, de manière à ce qu'une même ressource ne soit pas affectée à deux tâches concomitantes. Plusieurs variantes intéressantes sont répertoriées. Dans une première variante, chaque tâche doit être affectée et il s'agit alors de minimiser le coût de cette affectation, alors qu'une deuxième variante correspond au cas où les ressources sont limitées, il s'agit alors de maximiser le gain correspondant aux tâches affectées. Deux autre variantes, moins intéressantes dans le cadre de notre étude, correspondent respectivement au cas où les tâches ont une durée fixée, mais peuvent commencer à plusieurs instants, et au cas où les tâches peuvent être préemptées.

Le FJSP et l'ISP étant équivalents [126], nous emploierons indifféremment l'appellation FJSP pour ces deux problèmes. De manière générale, le FJSP considère implicitement des ressources machines et non des ressources humaines, ce qui signifie que les ressources sont généralement considérées comme étant disponibles de manière continue. À l'inverse, le PTSP s'intéresse au cas où chaque ressource est associée à une plage horaire de disponibilité, ce qui s'apparente à la notion de vacation dans le cadre de la gestion des ressources humaines. Le PTSP est un problème relativement récent, introduit par Krishnamoorthy *et al.* [127]. Ce problème regroupe en réalité plusieurs variantes identifiées au moyen d'une nomenclature faisant intervenir quatre éléments distincts : les spécificités des tâches (T) et celles des vacations (S), les qualifications des employés (Q) et la fonction objectif (O). Les différentes configurations envisagées par les auteurs pour ces quatre éléments sont données dans la Table 3.3. Grâce à cette nomenclature, il est alors possible de désigner précisément chaque variante du PTSP. Par exemple, la version de base du PTSP se note PTSP[F ;F ;H ;W], ce qui signifie que les tâches sont fixées dans le temps, que chaque ressource est associée à un ensemble connu et fixé de tâches affectables et que l'objectif est de minimiser le nombre de ressources permettant de couvrir toutes les tâches. Dans le cas particulier où toutes les ressources ont le même coût d'utilisation, cette variante est alors désignée sous l'intitulé *Shift Minimisation Personnel Task Scheduling Problem* (SMPTSP). Il s'agit de la variante la plus étudiée du PTSP [87, 127, 128, 140].

Nous abordons dans la section suivante les problèmes d'affectation de tâches fixées en distinguant deux variantes relativement courantes. La première variante correspond au cas où il s'agit d'affecter un maximum de tâches à un ensemble de ressources obligatoires, alors que la seconde s'intéresse au cas où il faut obligatoirement affecter toutes les tâches aux ressources et chercher à minimiser le coût de l'ensemble des ressources.

Attribut	Valeur	Description
T	F	Tâches fixées dans le temps et continues
	V	Tâches avec une durée variable
	S	Tâches non continues
	C	Temps de changement entre deux tâches consécutives
S	F	Vacations fixées dans le temps
	I	Vacations couvrant toutes les tâches
	D	Vacations non fixées avec une durée maximale
	U	Nombre illimité de vacations
Q	I	Qualifications identiques des employés
	H	Qualifications différentes des employés
O	F	Problème de faisabilité
	A	Minimiser le coût de l'affectation
	T	Minimiser le coût du travail
	W	Minimiser le nombre d'employés
	U	Minimiser le nombre de tâches non affectées

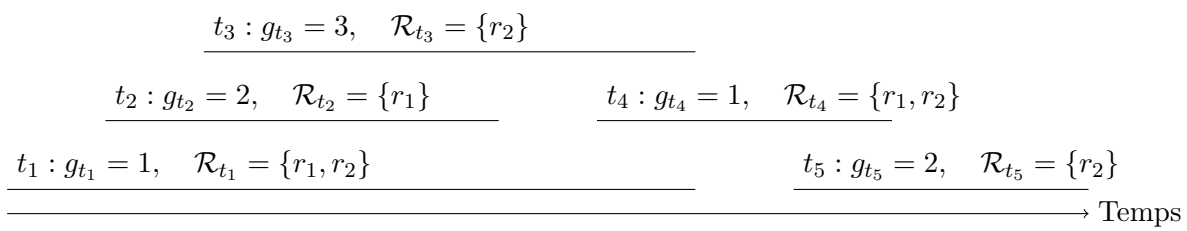
TABLE 3.3 – Nomenclature pour le PTSP.

3.2.1 Problèmes d'affectation avec ressources obligatoires

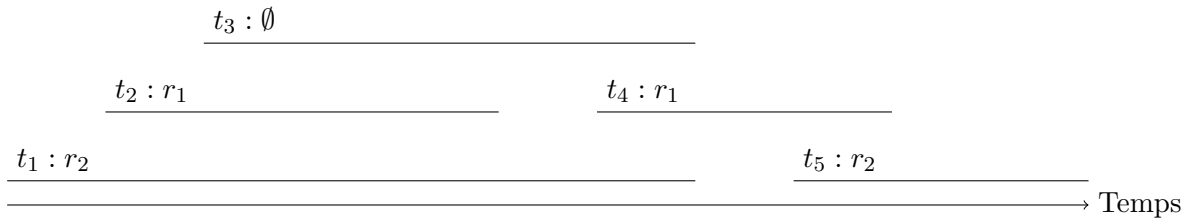
Il s'agit ici d'affecter un ensemble de tâches fixées dans le temps et non préemptables à un ensemble de ressources obligatoires. L'objectif est de maximiser le profit correspondant à l'affectation des tâches, sachant que deux tâches concomitantes ne peuvent pas être affectées à la même ressource et que chaque ressource ne peut effectuer qu'une partie des tâches. Les ressources sont dites obligatoires pour souligner le fait qu'on ne cherche pas à minimiser leur coût. Il s'agit donc d'un problème opérationnel que nous désignerons par l'acronyme OFJSP pour *Operationnal Fixed Job Scheduling Problem*. Plus formellement, considérons un ensemble de ressources $\mathcal{R}$ et un ensemble de tâches $\mathcal{T}$. Chaque tâche $t \in \mathcal{T}$ est associée à un gain g_t résultant de l'affectation de cette tâche. Chaque ressource $r \in \mathcal{R}$ ne peut réaliser qu'un sous-ensemble de tâches $\mathcal{T}_r \subset \mathcal{T}$. De manière symétrique, chaque tâche $t \in \mathcal{T}$ ne peut être réalisée que par un sous-ensemble de ressources $\mathcal{R}_t \subset \mathcal{R}$. Le problème OFJSP consiste alors à déterminer une affectation des tâches aux ressources qui maximise le gain total, sachant qu'une ressource donnée ne peut pas être affectée à deux tâches concomitantes. Une illustration de ce problème est donnée à la Figure 3.3. Sur cet exemple, deux ressources r_1 et r_2 sont disponibles pour 5 tâches $t_{1,..,5}$ (*cf.* Figure 3.3a). Chaque tâche est associée à un gain et ne peut être affectée qu'à un sous-ensemble de ressources. Par exemple, la tâche t_2 présente un gain de 2 et ne peut être affectée qu'à la ressource r_1. Une affectation réalisable de gain 6 est donnée à la Figure 3.3b. Nous pouvons constater que la tâche t_3 n'est pas affectée, car les deux seules ressources disponibles sont déjà assignées aux tâches t_2 et t_1 qui sont concomitantes avec t_3.

Quelques travaux portant sur OFJSP

Les premiers travaux portant sur l'OFJSP se sont tout d'abord intéressés au cas où toutes les ressources sont identiques, *i.e.* en mesure de réaliser toutes les tâches [32, 54]. Arkin et Silverberg [7] ainsi que Kroon *et al.* [129] étudient ensuite l'impact de la prise en compte des compétences. Par la suite, plusieurs travaux se sont intéressés au problème avec compétences [13, 122, 193]. Certains travaux différencient les machines selon d'autres critères comme par exemple leur vitesse de travail [21] ou leurs disponibilités [36]. Une autre variante du OFJSP consiste à prendre en compte des contraintes temporelles sur l'utilisation des ressources. Deux contraintes principales sont généralement considérées : les contraintes de charges [187] et les contraintes d'amplitudes [32, 187, 175]. Ces



(a) Données d'entrée sur les tâches : intervalle, gain et ressources possibles.



(b) Solution réalisable de gain total 6.

FIGURE 3.3 – Exemple de OFJSP.

contraintes sont particulièrement intéressantes dans le cadre de notre étude, car elles s'apparentent aux contraintes sur le temps présentiel et sur le temps travaillé. En revanche, les travaux prenant en compte de telles contraintes considèrent des ressources identiques.

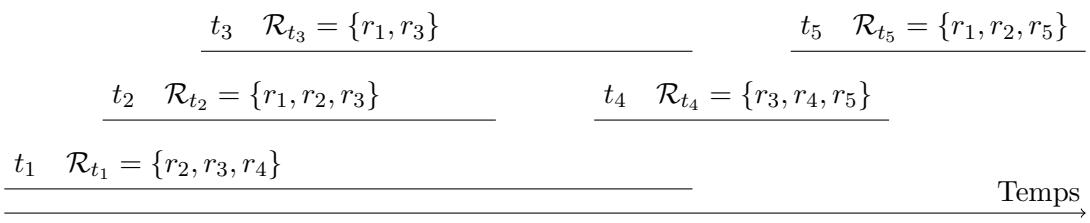
Différences et point communs avec le cas d'étude

Étant donné que notre problème porte en partie sur l'affectation de ressources à un ensemble de tâches fixées dans le temps, non préemptables et requérant des compétences particulières, il semble important de le positionner par rapport au problème OFJSP. Le fait de maximiser le profit réalisable par un ensemble de ressources correspond dans notre cas à rechercher une affectation avec un nombre minimum de tâches non affectées. Certaines variantes du OFJSP prennent en compte des contraintes d'amplitude et de charge qui correspondent aux contraintes de temps présentiel (travaillé et non travaillé) maximal. Autrement dit, certaines variantes du OFJSP s'apparentent à notre cas d'étude sur un horizon d'une journée. En effet, les contraintes de repos entre deux journées de travail ne sont pas abordées par les problèmes de type OFJSP. Pour résoudre notre problème, il est important de considérer l'horizon de planification dans son ensemble, de manière à couvrir toutes les tâches. Par conséquent, le fait que les contraintes de repos entre deux utilisations d'une même ressource ne soient pas abordées dans le cadre du OFJSP constitue une différence importante. D'autre part, l'objectif d'équité que nous souhaitons aborder ne correspond pas aux objectifs classiques du problème OFJSP.

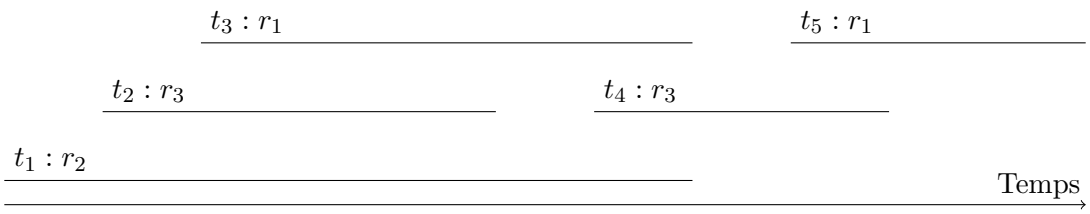
3.2.2 Problèmes d'affectation avec tâches obligatoires

Il s'agit ici d'affecter un ensemble de tâches fixées dans le temps et non préemptables à un ensemble de ressources de manière à ce que le coût des ressources utilisées soit minimal. Il s'agit donc d'un problème tactique que nous désignerons par l'acronyme TFJSP pour *Tactical Fixed Job Scheduling Problem*. En reprenant les notations d'OFJSP et en y ajoutant un coût d'utilisation c_r pour la ressource $r \in \mathcal{R}$, le problème TFJSP consiste alors à déterminer une affectation des tâches à un ensemble de ressources de coût minimal, sachant que chaque tâche doit être affectée à une ressource qualifiée et que les ressources ne peuvent pas être affectées à des tâches concomitantes. Une illustration de ce problème est donnée à la Figure 3.4. Sur cet exemple, cinq ressources $r_{1,..,5}$ de coût unitaire sont disponibles pour cinq tâches $t_{1,..,5}$, et chaque tâche ne peut être affectée qu'à un sous-ensemble de

ressources (*cf.* Figure 3.4a). Ainsi, par exemple, la tâche t_3 ne peut être affectée qu'aux ressources r_1 et r_3. Puisque le nombre maximal de tâches concomitantes est de trois, avec par exemple les tâches t_1, t_2 et t_3, il faut au moins trois ressources distinctes pour couvrir l'ensemble des tâches. Une affectation réalisable mobilisant trois ressources (donc optimale) est donnée à la Figure 3.4b.



(a) Données d'entrée.



(b) Solution réalisable mobilisant 3 ressources.

FIGURE 3.4 – Exemple de TFJSP.

Quelques travaux portant sur TFJSP

Plusieurs variantes du TFJSP ont été étudiées dans la littérature. Par exemple, Fischetti *et al.* étudient successivement le cas avec contraintes de charge [91] puis avec contraintes d'amplitude [90]. Pour ces deux travaux, les ressources sont identiques et il s'agit de minimiser le nombre de ressources utilisées. Kolen et Kroon [124], ainsi que Kroon *et al.* [130] évaluent quant à eux l'impact du relâchement de la contrainte de non préemption des tâches dans deux contextes différents : lorsque les ressources ont des intervalles de disponibilités différents et lorsque les ressources ont des compétences différentes. De manière similaire, Dondeti et Emmons étudient également le cas sans préemption [74], puis le cas avec préemption [75], selon différentes configurations de ressources (identiques ou différenciées). Le cas où les ressources ont des compétences différentes est par ailleurs fréquemment étudié [112, 123, 130]. Le cas particulier où les ressources ont des coûts unitaires ainsi que des compétences et des disponibilités horaires est abordé par le SMPTSP, introduit précédemment. Ce problème a récemment été étudié dans plusieurs travaux [86, 87, 127, 140, 182, 183]. Nous présentons au Chapitre 9 une méthode exacte basée sur un modèle de programmation par contraintes permettant de résoudre à l'optimum toutes les instances de la littérature dans un temps compétitif [87].

Différences et point communs avec le cas d'étude

De la même manière que pour l'OFJSP, le TFJSP ne prend pas en compte les contraintes de repos entre deux utilisations consécutives d'une même ressource, ce qui constitue en soit une différence importante. Puisque le TFJSP correspond à un problème tactique, il semble cohérent de le comparer à la version tactique de notre cas d'étude, qui correspond à la minimisation du coût total des intérimaires permettant de couvrir toutes les tâches. Dans la mesure où il s'agit d'un problème tactique, il est envisageable de considérer que les horaires du personnel sont fixés de manière à respecter automatiquement les temps de repos. Dans un tel contexte, le TFJSP se rapproche de notre cas d'étude. La

prise en compte des compétences des intérimaires reste cependant un problème majeure car il est très compliqué pour Biotrial de prévoir l'efficacité des différents intérimaires par rapport aux différentes tâches qui leur seront confiées. Par ailleurs le coût associé aux embauches d'intérimaires n'est pas aussi tranché que l'objectif de minimisation du nombre total de ressources. En effet, les intérimaires peuvent être embauchés sur une période plus ou moins longue, entraînant alors des coûts différents. De plus, il peut être intéressant d'embaucher un même intérimaire sur une période longue au lieu d'embaucher deux intérimaires sur des périodes courtes de manière à capitaliser sur plusieurs points tels que les formations, l'adaptation au rythme de travail et l'intégration aux équipes de travail.

3.3 Conclusion

Le problème que nous étudions se situe au point de rencontre de l'affectation de tâches fixées et de la planification de personnel. Nous avons par conséquent discuté des points communs et des différences entre notre cas d'étude et les différents problèmes de la littérature qui abordent ces deux grandes thématiques. Nous nous sommes tout d'abord intéressés au NRP qui aborde le problème de construction du planning des infirmiers, mais avec un nombre très limité de vacations disponibles. En réalité, le NRP s'intéresse davantage à la gestion des contraintes de séquences portant sur les plannings individuels des employés que sur leurs activités précises au sein de leurs journées de travail. Nous nous sommes ensuite intéressés au TSP qui se rapproche un peu plus de notre problème de construction des horaires de travail mais qui ne prend pas en compte la notion de tâche fixée dans le temps et non préemptable. Par conséquent, même si les horaires du personnel peuvent être construits avec une grande flexibilité, le TSP ne correspond pas non plus à notre cas d'étude. Ces deux problèmes de planification de personnel cherchent à couvrir une demande estimée en fonction de prévisions plus ou moins fiables. L'activité à couvrir est donc susceptible de varier au fil de la journée ce qui requiert une certaine flexibilité de la part des équipes de travail et donc une certaine flexibilité au niveau du planning du personnel. Pour cette raison, l'activité du personnel n'est jamais fixée avec le niveau de précision requis dans notre contexte. Notre cas d'étude nous amène en effet à considérer une demande parfaitement connue et complètement fixée dans le temps sur un horizon d'une semaine.

Contrairement au NRP et au TSP, le CPP considère une activité connue qui découle de trajets fixés dans le temps. L'objectif est alors de regrouper ces trajets de manière à former des rotations respectant des contraintes légales et organisationnelles, mais sans affecter ces rotations aux employés. Autrement dit, les contraintes de compétences, de disponibilités et de création d'un planning individuel ne sont pas prises en compte. Le CRP vise quant à lui à affecter des rotations à des employés qualifiés et disponibles de manière à construire des plannings individuels. En revanche, les journées de travail des employés sont déjà construites, et il s'agit uniquement de les affecter. Le problème intégré du CPP et CRP est donc très proche de notre cas d'étude, puisqu'il aborde les contraintes de compétences, de disponibilités, de non ubiquité, de non préemption, ainsi que les contraintes légales sur la construction des horaires de travail et leurs enchaînements. Cependant, en raison du contexte métier sous-jacent, plusieurs contraintes du CPP et du CRP n'apparaissent pas dans notre cas d'étude (cohérence géographique, repos hors base, changements de fuseaux horaires, *etc.*). Notre objectif d'équité est également très différent des objectifs traditionnels des problèmes de planification d'équipages.

Pour étudier plus particulièrement la problématique d'affectation de tâches fixées dans le temps, nous nous sommes intéressés à plusieurs problèmes de la littérature portant sur le sujet tels que le PTSP et le FJSP. Ces différents problèmes correspondent bien à notre problématique d'affectation de tâches, mais ils occultent le problème de construction des horaires du personnel. En effet, même si certaines variantes considèrent des contraintes intéressantes de charge maximale et d'amplitude maximale, les contraintes de repos entre deux périodes d'activité ne sont jamais prises en compte.

La Table 3.4 propose une vue d'ensemble des différents problèmes mentionnés dans ce chapitre, permettant ainsi de mieux comprendre le positionnement de notre cas d'étude. Les travaux y sont classés selon deux axes principaux. Un premier axe correspond au contenu des périodes de travail des ressources. Plus précisément, nous distinguons le cas où le contenu est non spécifié ou unique (NRP et certains TSP), du cas où les ressources réalisent des activités différentes durant une même période de travail. Ce deuxième cas est lui même affiné pour distinguer les activités préemptables (TSP) de celles qui ne le sont pas (PTSP, FJSP, CPP et CRP). Le deuxième axe porte sur les périodes de travail des ressources. Nous distinguons tout d'abord le cas où les périodes sont non spécifiées ou données (FJSP et PTSP) du cas où les périodes doivent être construites sur la base de contraintes. Ces contraintes peuvent porter sur la conception de l'horaire en lui même (contraintes intra-horaire) ou sur les séquences d'horaires acceptables (contraintes inter-horaire), ou sur ces deux cas. Par exemple, le CPP correspond au cas des contraintes intra-horaire car il s'agit de construire des rotations (*i.e.* des horaires de travail journaliers contraints), alors que le CRP correspond au cas des contraintes inter-horaire car il s'agit de construire des plannings individuels à partir des rotations. Notons que certains travaux apparaissent dans plusieurs configurations, principalement en raison des contraintes de non préemption. En effet, certains auteurs étudient des variantes d'un même problème avec et sans préemption, alors que d'autres étudient des problèmes où une partie de l'activité peut être préemptée, mais pas l'autre.

Le problème étudié dans cette thèse correspond à la configuration où l'on chercher à construire des périodes de travail comprenant des activités non préemptables tout en respectant des contraintes inter et intra-horaires. Par rapport aux autres configurations, il s'agit d'un sujet relativement peu étudié, ce qui souligne son intérêt académique. Les autres travaux considérant également des activités non préemptables ainsi que des contraintes intra et inter-horaires portent sur le problème intégré du CPP et du CRP, soit dans les transports aériens, soit dans les transports ferroviaires [52, 82, 96]. La résolution intégrée de ces deux problèmes permet d'obtenir de meilleures solutions, ce qui souligne l'intérêt d'une approche intégrée. Néanmoins, ces travaux considèrent des trajets alors que notre étude porte sur des tâches ce qui induit des différences au niveau des contraintes métier ainsi qu'au niveau des données. Par ailleurs, l'objectif des approches portant sur le CRP est généralement de minimiser le nombre d'employés permettant de couvrir tous les trajets, conduisant ainsi à utiliser chaque ressource au maximum de ses disponibilités, alors que nous cherchons ici à obtenir des plannings équitables vis-à-vis de la distribution de la charge globale en prenant en compte à la fois des tâches médicales et obligatoires, mais également des activités administratives non fixées et préemptables pour lesquelles il est néanmoins nécessaire de réserver un certain volume horaire.

Puisqu'aucun des problèmes présentés ne correspond tout à fait à notre cas d'étude, il nous semble pertinent de proposer des méthodes de résolution dédiées au contexte particulier de Biotrial. De plus, il ne nous semble pas possible de prendre en compte tout le contexte métier de Biotrial au moyen d'une méthode de la littérature.

Périodes de travail des ressources		Activités des ressources durant une période de travail		
		Non spécifiées ou uniques	Préemptables	Non-préemptables
Contraintes	Inter & Intra	[40, 76, 111, 120, 143]	[67, 84, 110, 132, 141, 174]	**Thèse**, [52, 82, 96]
	Inter	[35, 41, 43, 49, 89, 149, 159]	[72]	[53, 51, 69, 94, 144, 146, 196]
	Intra	[39, 148, 178]	[32, 33, 153]	[32, 33, 37, 90, 91, 175, 187, 196]
Données		[158]	[32, 33, 75, 80, 117, 124, 130, 137, 138]	[7, 13, 32, 33, 54, 74, 80, 87, 95, 112, 117, 122, 123, 124, 127, 129, 130, 138, 140, 182, 183, 193]

TABLE 3.4 – Positionnement de notre cas d'étude dans la littérature.

4

Formalisation du problème

Dans ce chapitre, nous formalisons le problème étudié et proposons une notation permettant de distinguer les différentes variantes étudiées durant cette thèse. Nous proposons également des algorithmes de pré-traitement et de post-traitement, ainsi qu'une méthode de calcul d'une borne inférieure pour l'une des variantes étudiées. Nous donnons une modélisation sous la forme d'un Programme Linéaire en Nombres Entiers qui cherche à affecter toutes les tâches de manière à minimiser l'inéquité entre les employés. À partir de ce premier modèle, nous en introduisons un second qui vise quant à lui à minimiser le nombre de tâches non affectées. Pour finir, nous proposons une version simplifiée de ce dernier modèle en supprimant la contrainte relative à la pause déjeuner de manière à mieux passer à l'échelle et ainsi obtenir davantage de bornes inférieures. Une étude expérimentale permet d'évaluer l'intérêt des modèles et des algorithmes donnés dans ce chapitre.

Sommaire

4.1 Définition du problème

4.1.1 Notations générales

Les notations du problème sont récapitulées au niveau des Tables 1 et 2, p. 3. Nous noterons respectivement $\mathcal{N}$, $\mathcal{T}$, $\mathcal{M}$ et $\mathcal{D}$ l'ensemble des employés, des tâches médicales, des tâches obligatoires et des jours de la semaine. L'union des ensembles $\mathcal{T}$ et $\mathcal{M}$ sera notée $\mathcal{E}$. Afin de modéliser la contrainte d'affectation des tâches de manière souple, nous introduisons un employé fictif, noté $N+1$ auquel il est possible d'affecter toutes les tâches, sans que cela ne viole aucune contrainte. L'ensemble des employés, complété de l'employé fictif, sera noté $\mathcal{N}^+$.

Puisque toutes les tâches sont supposées entièrement fixées (*cf.* R 3, p. 32) il est possible d'obtenir l'ensemble des ensembles maximaux de tâches concomitantes en temps polynomial (*cf.* Section 4.2.1, p. 64). En effet, cela revient à trouver l'ensemble des cliques maximales dans le graphe d'intervalle correspondant aux tâches [100], ce qui peut être réalisé en temps polynomial. Cet ensemble est noté $\mathcal{C}$. Pour toute tâche (médicale ou obligatoire) $t \in \mathcal{E}$, nous noterons respectivement $start_t$, end_t, $proc_t$ et day_t les dates de début et de fin, la durée et le jour de travail, déterminé en fonction de la date de début. Plus précisément, les tâches commençant entre minuit et six heures du matin sont rattachées à la journée précédente. Étant donné que le travail des employés peut s'effectuer à cheval sur deux jours calendaires, nous avons adopté un axe des temps unique sur toute la semaine, avec une granularité d'une minute, ce qui permet de positionner les tâches avec toute la précision requise (*cf.* Section 2.1.2, p. 24).

En ce qui concerne les employés, remarquons tout d'abord que les compétences et les disponibilités horaires d'un employé n permettent d'obtenir directement l'ensemble des tâches qui lui sont potentiellement assignables. Cet ensemble sera noté $\mathcal{T}_n$. Nous noterons également $\mathcal{M}_n$ l'ensemble des tâches obligatoires de l'employé n. L'union des ensembles $\mathcal{T}_n$ et $\mathcal{M}_n$ est noté $\mathcal{E}_n$. Les éléments de ces différents ensembles étant fixés dans le temps, il est donc possible de les regrouper par jour de travail. Puisque chaque élément appartient à un seul jour de travail, déterminé en fonction de sa date de début, nous obtenons ainsi une partition des ensembles $\mathcal{T}_n$, $\mathcal{M}_n$ et $\mathcal{E}_n$, notée respectivement $\mathcal{T}_{n,d}$, $\mathcal{M}_{n,d}$ et $\mathcal{E}_{n,d}$. La charge médicale idéale de l'employé n est notée $target_n$. Afin de prendre en compte le planning de la semaine passée de chaque employé, nous utilisons également les données suivantes :

$work_n^{-1}$ Le positionnement de la dernière minute de travail par rapport au dimanche de la semaine passée à minuit. Une valeur de 30, par exemple, signifie que l'employé n a travaillé jusqu'à 23h30 dimanche de la semaine passée.

off_n^{-1} Le positionnement du dernier jour de repos par rapport au lundi de la semaine courante. Une valeur de 2, par exemple, signifie que le weekend précédent a été travaillé, ce qui signifie que le dernier repos correspond au vendredi.

$break_n^{-1}$ Le positionnement en minutes de la fin du dernier repos hebdomadaire par rapport au lundi de la semaine courante 0h00. Une valeur de 1440 (24h), par exemple, signifie que l'employé n a terminé son dernier repos hebdomadaire samedi de la semaine passée à minuit.

4.1.2 Contraintes du problème

Les hypothèses de travail données à la Section 2.2.2, p. 29, nous conduisent à considérer à la fois des **contraintes organisationnelles** et des **contraintes légales**. Les premières correspondent essentiellement au problème d'affectation de tâches, alors que les secondes portent sur le problème de

construction des horaires de travail.

Contraintes organisationnelles :
C 1: Un employé ne peut réaliser qu'une seule vacation par journée de travail.
C 2: Toutes les tâches pré-assignées à un employé doivent lui être assignées.
C 3: Seules les tâches assignables à un employé peuvent lui être assignées.
C 4: Un employé ne peut traiter au plus qu'une seule tâche à la fois.
C 5: Une tâche est affectée à au plus un employé.
C 6: Une tâche ne peut pas être préemptée.
C 7: Une vacation ne peut contenir deux tâches appartenant à des journées de travail différentes.

Contraintes légales :
C 8: Le temps présentiel d'une vacation est inférieur à SPAN_{MAX}.
C 9: Le temps travaillé d'une vacation est inférieur à $\text{WORKD}_{\text{MAX}}$.
C 10: Le temps travaillé hebdomadairement par un employé est inférieur à $\text{WORKW}_{\text{MAX}}$.
C 11: Deux vacations d'un même employé sont séparées par un repos minimum de $\text{BREAKD}_{\text{MIN}}$.
C 12: Sur 7 jours glissants, un employé bénéficie d'une période de repos de $\text{BREAKW}_{\text{MIN}}$.
C 13: Sur 7 jours glissants, un employé bénéficie d'un jour de repos.
C 14: Les vacations du jour d d'une durée supérieure à SMALLD, commençant avant LUNCHS_d et terminant après LUNCHE_d comprennent un temps de repos cumulé d'une durée de LUNCHP.

Dans la suite de cette thèse, nous reprenons une notation sous la forme classique $\alpha|\beta|\gamma$ de manière à désigner explicitement les différentes variantes du problème étudié. Le cœur du problème est noté SDPTSP, pour *Shift Design Personnel Task Scheduling Problem*. Cet acronyme, utilisé dans le champ α, permet de positionner notre problème par rapport au PTSP tout en mentionnant clairement la construction des vacations.

4.1.3 Gestion de l'objectif

Modélisation du critère de réalisabilité

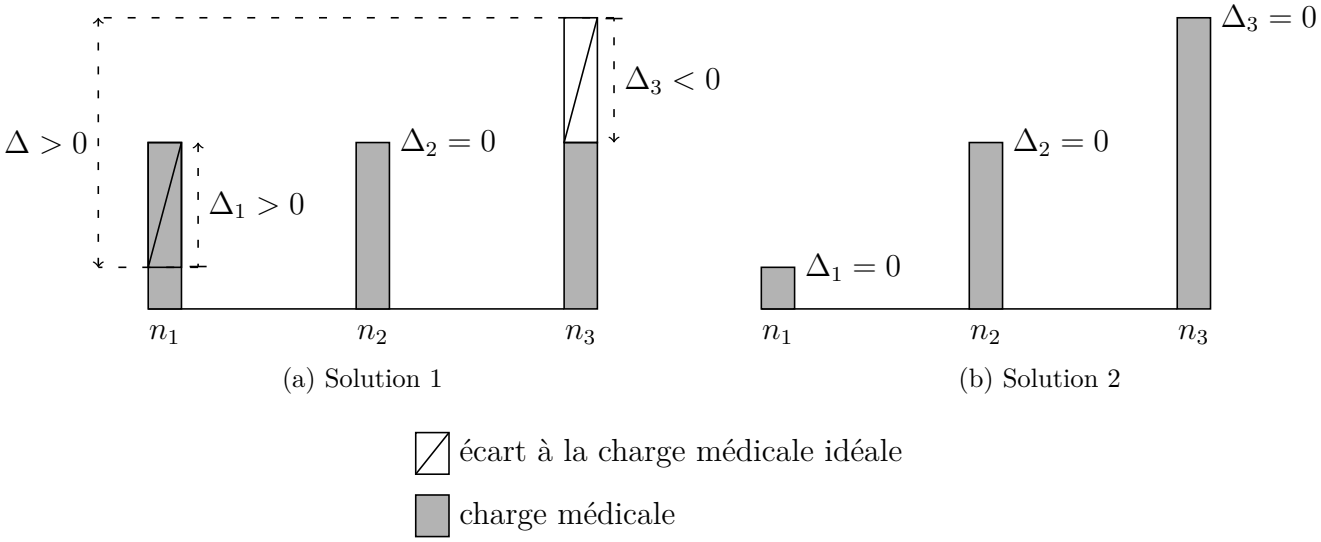
Les observations réalisées à la Section 2.2.3, p. 30, nous ont conduit à supprimer la contrainte métier stipulant que toutes les tâches doivent être affectées à un employé compétent et disponible. L'objectif principal de Biotrial reste cependant de construire des plannings pour lesquels toutes les tâches sont affectées, de manière à réduire le coût des intérimaires. Il s'agit donc de minimiser le nombre de tâches non affectées, noté U. Si l'instance est réalisable sans intérimaire, il existe alors au moins une solution pour laquelle U = 0. Dans ce cas, nous parlerons de **solutions complètes**, par opposition aux **solutions partielles**. Le problème consistant à trouver une solution respectant les contraintes C 1 à C 14 et qui minimise U, est noté SDPTSP||U.

Solution complète : une solution au problème SDPTSP|U=0|-

Solution partielle : une solution au problème SDPTSP|U>0|-

Modélisation du critère d'équité

En ce qui concerne la modélisation du critère d'équité, le but du décideur est de réduire au maximum la différence de traitement entre les employés. De cette manière, en période d'activité intense, la surcharge de travail est répartie sur tous les employés, alors qu'en période creuse, tous les employés profitent de la baisse d'intensité du travail. Dans le cas général, tous les employés doivent avoir assez de temps pour réaliser leurs activités administratives et obligatoires. Nous proposons par conséquent de minimiser l'écart entre le plus grand et le plus petit écart à la charge médicale idéale des employés.



(a) Solution 1 (b) Solution 2

FIGURE 4.1 – Exemple d'une distribution de la charge médicale

Pour un employé $n \in \mathcal{N}$, l'écart à la charge médicale idéale, noté Δ_n, est défini comme la différence entre la charge médicale affectée à l'employé et sa charge médicale idéale. Une petite différence entre le plus grand et le plus petit écart signifie que la charge de travail est bien répartie entre les différents employés. Plus formellement, pour un ensemble d'employés $\mathcal{N}$, chaque employé $n \in \mathcal{N}$ étant associé à une charge médicale idéale $target_n$ et à une charge médicale affectée W_n, l'inéquité Δ du planning correspondant est donnée par les contraintes (4.1) et (4.2).

$$\Delta = \max_{n \in \mathcal{N}} (\Delta_n) - \min_{n \in \mathcal{N}} (\Delta_n) \tag{4.1}$$

$$\forall n \in \mathcal{N}, \Delta_n = W_n - target_n \tag{4.2}$$

La Figure 4.1 propose une illustration de cette fonction objectif. La Figure 4.1a présente trois employés (n_1, n_2 et n_3) affectés à la même charge de travail médical ($W_1 = W_2 = W_3$). Cependant, ces employés n'ont pas la même charge médicale idéale ($target_1 < target_2 < target_3$). Par conséquent, n_1 est en surcharge par rapport à $target_1$, alors que n_3 est en sous-charge par rapport à $target_3$, ce qui signifie que l'écart à l'objectif de travail médical de l'employé n_1, Δ_1, est strictement positif, alors que l'écart à la charge médicale idéale de n_3, Δ_3, est strictement négatif. Par conséquent, l'inéquité associée à cette répartition du travail est strictement positive ($\Delta = \Delta_1 - \Delta_3 > 0$). A contrario, la Figure 4.1b, présente des employés réalisant exactement leur objectif de travail médical, ce qui correspond à une solution optimale ($\Delta = 0$). Le problème consistant à trouver une solution complète minimisant Δ sera noté SDPTSP|U = 0|Δ.

Modélisation de l'objectif global

L'objectif principal de Biotrial est d'obtenir une solution complète et, si possible, équitable vis-à-vis de la répartition de la charge de travail. S'il n'est pas possible d'obtenir une solution complète, il est tout de même intéressant de construire une solution équitable, car il est possible que le décideur soit en mesure de compléter la solution partielle en violant certaines contraintes. Dans ce cas, il est préférable que la solution partielle soit équitable pour que les employés acceptent les modifications proposées par le décideur. Cependant, il est d'autant plus facile de compléter une solution partielle

que celle-ci présente peu de tâches non affectées. Autrement dit, il s'agit de minimiser le nombre de tâches non affectées et, si possible, minimiser l'inéquité entre les employés. Par conséquent, nous considérons les objectifs U et Δ de manière lexicographique. De cette manière une solution avec un plus petit nombre de tâches non affectées est préférée à une solution plus équitable, alors que deux solutions présentant le même nombre de tâches non affectées seront départagées par rapport au critère d'équité. Le fait de tenir compte de ces deux objectifs nous permet d'améliorer l'équité d'une solution, même lorsqu'il n'est pas possible d'obtenir une solution complète. Le problème consistant à trouver une solution respectant les contraintes C 1 à C 14 et minimisant les objectifs U et Δ de manière lexicographique est noté SDPTSP||U,Δ.

4.1.4 Représentation d'une solution

Puisque toutes les tâches sont fixées dans le temps, une solution correspond directement à une affectation des tâches aux employés. Tout d'abord, à partir d'une affectation tâche-employé, il est possible de vérifier les contraintes organisationnelles et il est possible de déduire les horaires de travail des employés, ce qui permet de vérifier les contraintes légales. Autrement dit, l'affectation tâche-employé permet de vérifier toutes les contraintes. L'affectation tâche-employé permet également d'obtenir le nombre de tâches non affectées (U) ainsi que l'inéquité entre les employés (Δ), ce qui permet de calculer le coût de la solution. Pour finir, une affectation tâche-employé ne correspond qu'à une seule solution. Par conséquent, les solutions seront identifiées par leur affectation.

Puisqu'une solution se représente naturellement via son affectation, il semble cohérent d'opter pour une modélisation de type affectation au lieu d'une modélisation indexée sur le temps. De cette manière, la contrainte de non préemption peut être traitée de manière implicite, alors qu'une modélisation indexée sur le temps nous conduirait à vérifier qu'une tâche commencée par un employé est réalisée de bout en bout par ce même employé. Cela est d'autant plus décisif que le problème de planification de Biotrial s'étend sur un horizon d'une semaine avec une granularité d'une minute.

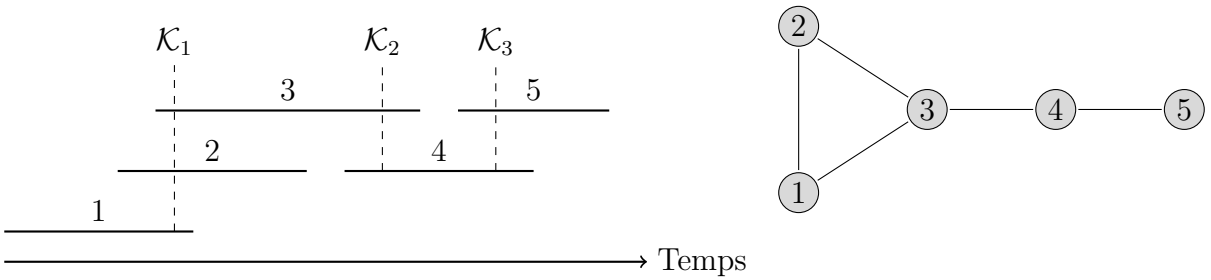
4.1.5 Variantes étudiées

Les différentes variantes connexes au SDPTSP et intéressantes dans le cadre de notre étude sont données ci-dessous. Par abus de langage, nous parlerons d'instances irréalisables pour désigner les instances qui n'admettent aucune solution complète, ce qui correspond à une irréalisabilité du problème SDPTSP|U=0|-. Dans cette thèse, nous nous concentrerons principalement sur SDPTSP||U,Δ.

Nom	Description
SDPTSP||U	Trouver une solution respectant les contraintes C 1 à C 14 et minimisant U.
SDPTSP|U = 0|Δ	Trouver une solution respectant les contraintes C 1 à C 14, avec une affectation complète et minimisant Δ.
SDPTSP||U,Δ	Trouver une solution respectant les contraintes C 1 à C 14 et minimisant lexicographiquement U et Δ.

4.2 Pré/Post traitements et calcul d'une borne inférieure

Nous rappelons tout d'abord l'algorithme permettant d'obtenir l'ensemble des cliques maximales dans un graphe d'intervalles et montrons comment utiliser cet algorithme en pré-traitement du SDPTSP. Par la suite, nous proposons une borne inférieure pour SDPTSP||U, ainsi qu'un algorithme de post-traitement, H_Δ, permettant de réduire l'inéquité entre les employés.



(a) Distribution des tâches. (b) Graphe d'intervalles.

FIGURE 4.2 – Exemple d'une recherche des cliques maximales dans un graphe d'intervalles

4.2.1 Ensemble des cliques maximales dans un graphe d'intervalles

Considérons un ensemble de tâches, $\mathcal{T}$, fixées dans le temps. Chaque tâche représente alors un intervalle $[s_i; e_i]$, et le graphe d'intervalles correspondant s'obtient en associant un sommet à chaque tâche, et en reliant les sommets correspondant à des intervalles concomitants. Dans le contexte de notre cas d'étude, la notion de graphe d'intervalles est très intéressante car elle permet de représenter efficacement le positionnement relatif des tâches. Plus précisément, remarquons qu'un ensemble de tâches concomitantes se traduit au niveau du graphe d'intervalles par une clique. Par conséquent, la recherche de l'ensemble des cliques maximales dans un graphe d'intervalles permet de trouver l'ensemble des ensembles maximaux de tâches concomitantes. La Figure 4.2 illustre cela sur un ensemble de 5 tâches, conduisant à 3 cliques maximales $\mathcal{K}_1 = \{1, 2, 3\}$, $\mathcal{K}_2 = \{3, 4\}$ et $\mathcal{K}_3 = \{4, 5\}$. Puisqu'un employé ne peut pas travailler sur plus d'une tâche à la fois (C 4), et puisqu'il est interdit de préempter les tâches (C 6), toutes les tâches d'une même clique doivent être affectées à des employés différents. Par conséquent, le fait de connaître les cliques maximales permet d'exprimer efficacement la contrainte C 4. Dans le cas général, la recherche des cliques maximales dans un graphe est un problème NP-Difficile, mais dans le cas d'un graphe d'intervalles, cela peut être réalisé en temps polynomial [101], au moyen de l'Algorithme 4.1. Puisque toutes les tâches sont fixées dès le début du problème, cet algorithme est utilisé une seule fois en pré-traitement. De cette manière, la contrainte C 4 peut être gérée de manière efficace durant toute la phase de résolution.

4.2.2 Borne inférieure pour SDPTSP||U

Comme mentionné à la Section 2.2.3, il est possible que l'effectif de travail ne puisse pas couvrir l'ensemble des tâches, ce qui signifie que SDPTSP|U = 0|Δ est alors infaisable. Généralement, il est difficile d'identifier la raison de cette infaisabilité, car de nombreuses données rentrent en compte : législation, compétences, disponibilités, positionnement des tâches, *etc.* Le cas particulier des pics d'activité est en revanche plus simple à expliquer, puisqu'il correspond à un manque d'effectif sur une période donnée. Ce manque peut découler du positionnement des horaires de travail, mais il est possible que l'effectif disponible, indépendamment des horaires de travail, soit simplement insuffisant. Nous nous intéressons ici à la détection de ces infaisabilités. Pour cela, nous proposons une méthode utilisant le concept de profil de charge pour obtenir une borne inférieure pour SDPTSP||U (*cf.* Algorithme 4.2). Le profil de charge donne l'ensemble des tâches concomitantes à chaque intervalle horaire, *i.e.* un intervalle formé de deux événements consécutifs, comme le début ou la fin d'une tâche. Une illustration de cette notion de profil de charge est donnée à la Figure 4.3. La borne inférieure que nous proposons, notée $\underline{U}$, correspond à la plus grande différence sur l'ensemble des intervalles, entre le nombre de tâches à couvrir et le nombre d'employés disponibles. Nous considérons ici qu'un employé est disponible sur un intervalle si cet intervalle n'est ni concomitant à l'une de ses indisponibilités, ni concomitant à son repos hebdomadaire (quand celui-ci est connu). Les instances vérifiant $\underline{U} \geq 1$ sont alors infaisables pour SDPTSP|U = 0|Δ.

Algorithme 4.1: Recherche des cliques maximales dans un graphe d'intervalles

Données :
$\mathcal{I}$, liste d'intervalles $[s_i; e_i]$

fonction `trouverCliquesMax`($\mathcal{I}$)
  $k \leftarrow 0$
  continuer $\leftarrow$ vrai
  $S \leftarrow \min_{i \in \mathcal{I}} s_i$
  $\mathcal{C} \leftarrow \emptyset$
  tant que *(continuer)* **faire**
    $E \leftarrow \min_{i \in \mathcal{I}, e_i > S} e_i$
    $k \leftarrow k + 1$
    $\mathcal{K}_k \leftarrow \{i \in \mathcal{I}; s_i < E \leq e_i\}$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{K}_k$
    si $(\exists s_i \geq E)$ **alors**
      $S \leftarrow \min_{i \in \mathcal{I}, s_i \geq E} s_i$
    sinon
      continuer $\leftarrow$ faux
  retourner $\mathcal{C}$

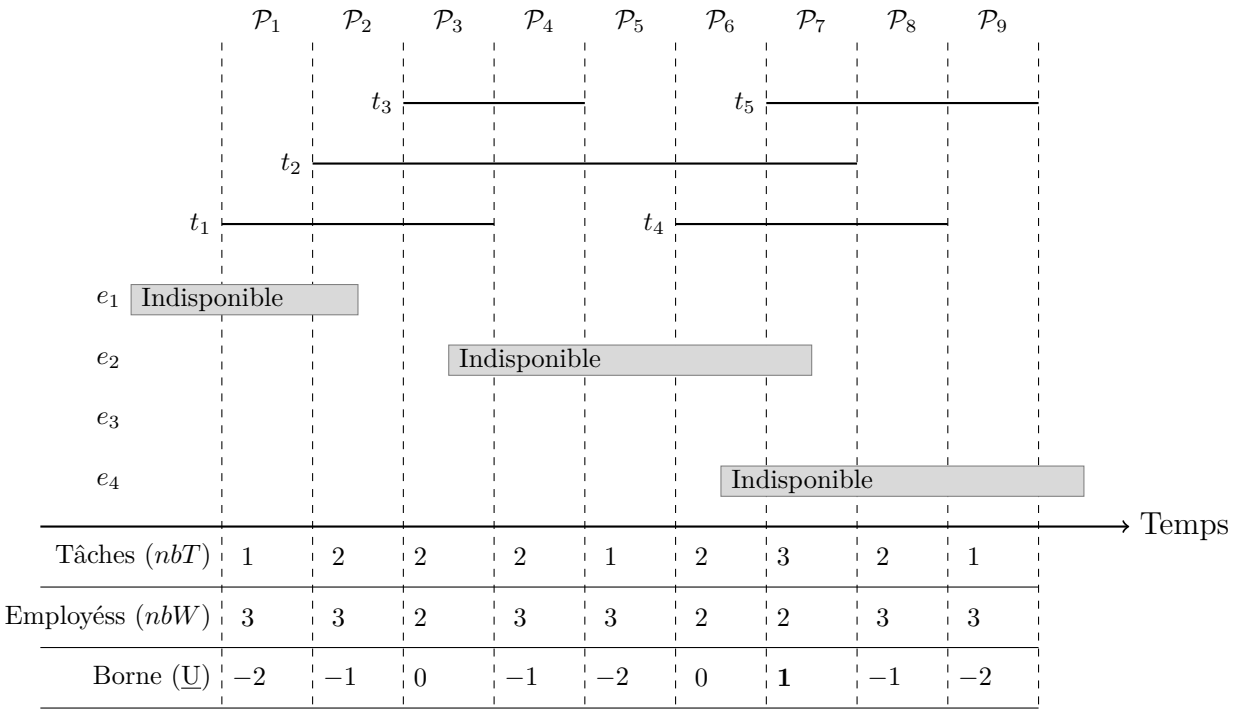


FIGURE 4.3 – Profil de charge & borne inférieure sur le nombre de tâches non affectées

Algorithme 4.2: $H_{\underline{U}}$: recherche d'une borne inférieure pour U

Données :
SDPTSP, une instance du SDPTSP

```
1  fonction H_U(SDPTSP)
2      P ← genererProfil(SDPTSP)
3      P ← card(P)
4      U ← 0
5      pour (i = 1,..,P) faire
6          nbT ← card(P_i)
7          nbW ← nbEmployésDisponible(P_i, SDPTSP)
8          si (nbT − nbW > U) alors
9              U ← nbT − nbW
10     retourner U
```

4.2.3 H_{Δ}, une descente locale pour réduire Δ

L'inéquité d'une solution, donnée à la Section 4.1.3, se déduit de l'affectation des tâches aux employés et de leurs charges médicales idéales. A priori, seul un nombre réduit d'employés intervient effectivement dans le calcul de cet objectif : les employés qui sont le plus au-dessus de leur charge idéale et ceux qui sont le plus en dessous. Par ailleurs, les activités administratives n'étant pas fixées dans le temps, le planning des employés est relativement flexible. Par conséquent, il est aisé de déplacer les tâches d'un employé à un autre ou de réaliser des échanges de tâches entre les employés. Dans un tel contexte, la réaffectation d'une tâche depuis un employé très au-dessus de sa charge cible vers un employé très en dessous de sa charge cible peut conduire à réduire efficacement l'inéquité d'une solution.

Pour réduire l'inéquité Δ d'une solution, nous proposons une descente locale, notée H_{Δ} (*cf.* Algorithme 4.3). Les horaires du personnel sont tout d'abord complétés et étendus au maximum de manière à gagner en flexibilité vis-à-vis des réaffectations envisageables. Par la suite, les horaires de travail ne sont plus modifiés et nous réduisons la valeur de l'inéquité en réaffectant et en échangeant des tâches. Cela se fait au moyen de quatre étapes répétées jusqu'à obtention d'un minimum local. Nous proposons de réaffecter en priorité les tâches les plus longues de manière à réaliser en premier les réaffectations les plus significatives. Une première étape consiste à sélectionner l'employé le plus en dessous de sa charge idéale pour lui assigner l'une des tâches affectée à un autre employé, de manière à ce que cette réaffectation réduise l'inéquité globale (*cf.* Figure 4.4). Symétriquement, une deuxième étape consiste à sélectionner l'employé le plus au-dessus de sa charge idéale pour réaffecter l'une de ses tâches à un autre employé. Ces deux premières étapes permettent de réduire l'inéquité en réaffectant certaines tâches, mais elles ne permettent pas de gérer efficacement les réaffectations de tâches concomitantes. Si on souhaite déplacer la tâche t_1 depuis l'employé e_1 vers l'employé e_2 et la tâche t_2, depuis l'employé e_2 vers l'employé e_1, il faut alors que e_2 soit entièrement disponible durant la durée de t_1 et réciproquement, que e_1 le soit durant la durée de t_2. Si t_1 et t_2 sont concomitantes, alors il faut nécessairement passer par un troisième employé « temporaire ». Cela n'est pas très efficace car il faut alors trois employés au lieu de deux. De plus, si le mouvement intermédiaire dégrade l'objectif, il ne sera pas retenu. Pour corriger cela, nous proposons d'appliquer le principe de ces deux premières étapes dans le cadre d'un échange de deux affectations au lieu d'une simple réaffectation (*cf.* Figure 4.5). Cela accroît la marge de manœuvre lors de la réaffectation des tâches, ce qui permet de réduire l'inéquité alors qu'une simple réinsertion fonctionnerait moins bien.

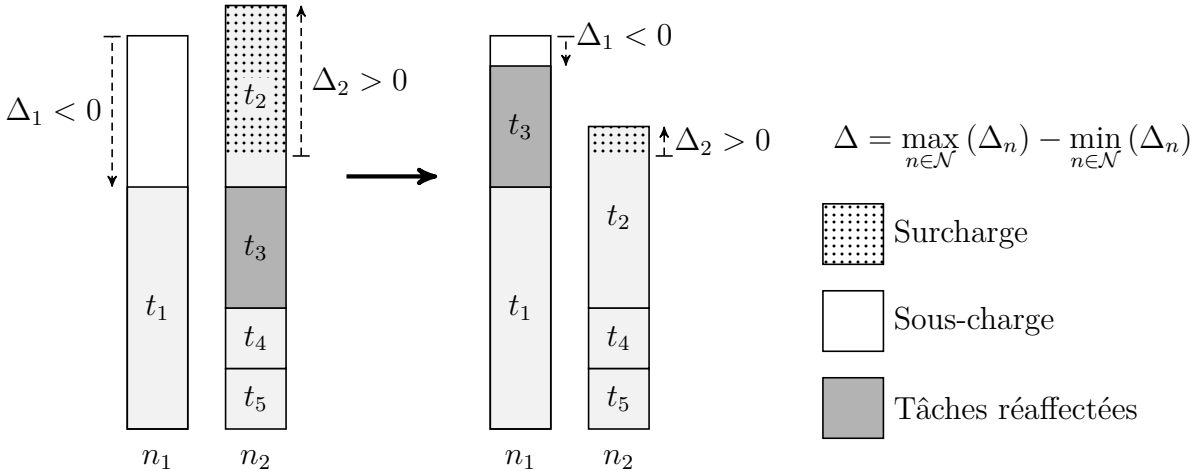


FIGURE 4.4 – Réduction de l'inéquité via H_Δ, cas d'une insertion

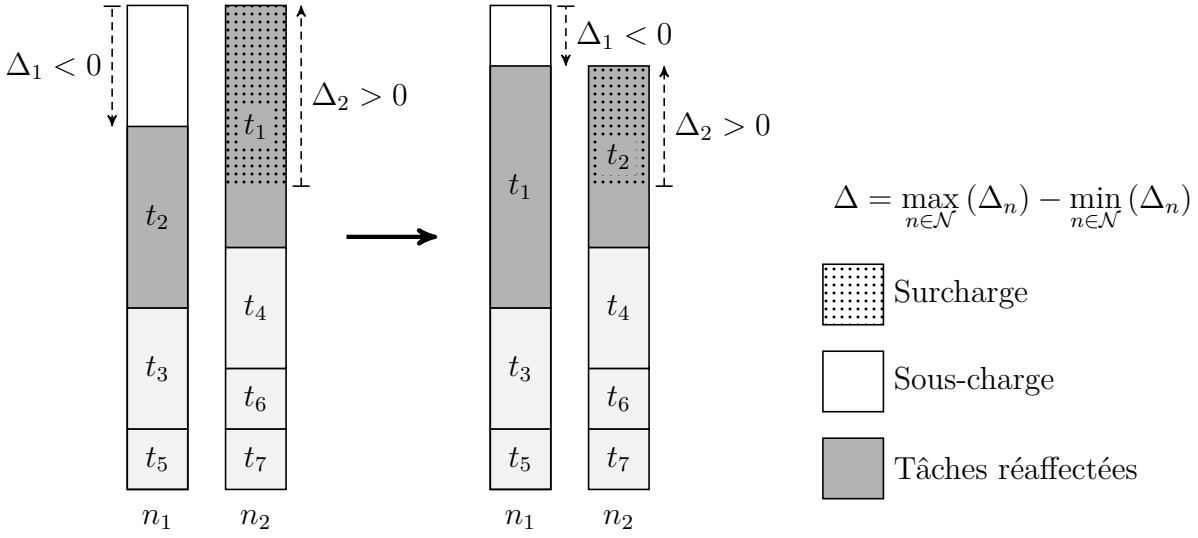


FIGURE 4.5 – Réduction de l'inéquité via H_Δ, cas d'un échange

Quelques propriétés de H_Δ

Bien que cela semble évident, remarquons tout d'abord que H_Δ ne donne pas la valeur optimale de Δ. Remarquons également qu'il est possible de continuer à réduire Δ en appliquant H_Δ plusieurs fois de suite. En effet, la première étape de H_Δ consiste à étendre aléatoirement les horaires de travail des employés à partir de l'affectation courante, ce qui peut conduire à « débloquer » la descente locale. Pour finir, notons que H_Δ ne conserve par l'ordre des solutions. Autrement dit, soient deux solutions S_1 et S_2 présentant une inéquité respective Δ_1 et Δ_2, telles que $\Delta_1 < \Delta_2$. En appliquant H_Δ sur S_1 et S_2, nous obtenons alors deux nouvelles solutions $S_1^{'}$ et $S_2^{'}$ présentant une inéquité respective $\Delta_1^{'}$ et $\Delta_2^{'}$, telles que $\Delta_1^{'}$ n'est pas forcément inférieure à $\Delta_2^{'}$.

Algorithme 4.3: H_Δ : descente locale pour réduire l'inéquité

Données :
SDPTSP, une instance du SDPTSP
$\mathcal{H}$, les horaires du personnel
$\mathcal{A}$, une affectation des tâches aux employés

```
1  fonction H_Δ(SDPTSP, H, A)
2      H ← ajouterHoraires(H)
3      H ← etendreHoraires(H)
4      répéter
5          succes ← ajoutSousCharge(SDPTSP, H, A)
6      jusqu'à (¬succes)
7      répéter
8          succes ← retraitSurCharge(SDPTSP, H, A)
9      jusqu'à (¬succes)
10     répéter
11         succes ← echangeSousCharge(SDPTSP, H, A)
12     jusqu'à (¬succes)
13     répéter
14         succes ← echangeSurCharge(SDPTSP, H, A)
15     jusqu'à (¬succes)
16     retourner(H, A)
```

4.3 Modèles PLNE autour de SDPTSP

Nous proposons ici une modélisation du SDPTSP$|U = 0|\Delta$ sous la forme d'un PLNE, noté PLNE$_{|U=0|\Delta}$. Nous adaptons ensuite cette première modélisation au problème SDPTSP$||U$, dont la résolution nous permet d'évaluer la difficulté des instances proposées à la Section 2.3.2, p. 33. Cela nous permet également d'établir une base de travail à partir de laquelle nous pourrons évaluer d'autres méthodes de résolution.

4.3.1 Modélisation PLNE pour SDPTSP$|U = 0|\Delta$

Variables du modèle PLNE$_{|U=0|\Delta}$

Les variables du modèle PLNE$_{|U=0|\Delta}$ sont données à la Table 4.1. Ce modèle repose principalement sur les variables binaires $Assign_{t,n}$ qui prennent la valeur 1 si et seulement si la tâche t est affectée à l'employé n. Les variables Δ_n donnent l'écart à la charge médicale idéale de chaque employé, ce qui permet d'obtenir la valeur globale de l'inéquité Δ.

Les dates de début et de fin des vacations des employés, respectivement notées $First_{d,n}$ et $Last_{d,n}$, permettent de garantir les temps de repos minimum et les temps de travail maximum des employés. Les variables binaires $Lunch_{d,n}$ prennent la valeur 1 si et seulement si l'employé n doit avoir une pause déjeuner durant la vacation d. Le temps travaillé des employés, noté $Work_{d,n}$, est alors déduit des variables $First_{d,n}$, $Last_{d,n}$ et $Lunch_{d,n}$.

$\Delta \in \mathbb{N}$	Valeur de l'inéquité
$\forall n \in \mathcal{N}, \Delta_n \in \mathbb{Z}$	Écart à la charge médicale idéale de l'employé n
$\forall n \in \mathcal{N}, \forall t \in \mathcal{E}, Assign_{t,n} \in \{0,1\}$	$= 1$ ssi la tâche t est affectée à l'employé n
$\forall \boldsymbol{d \in \mathcal{D}}, \forall \boldsymbol{n \in \mathcal{N}},$	**Pour toute vacation d, pour tout employé n**
$First_{d,n} \in \mathbb{Z}$	Début de la vacation d pour n
$Last_{d,n} \in \mathbb{Z}$	Fin de la vacation d pour n
$Work_{d,n} \in \mathbb{N}$	Temps travaillé par n durant la vacation d
$Lunch_{d,n} \in \{0,1\}$	$= 1$ ssi n déjeune durant la vacation d
$Empty_{d,n} \in \{0,1\}$	$= 1$ ssi la vacation d de n est vide
$Lunch^-_{d,n} \in \{0,1\}$	$= 1$ ssi n commence la vacation d avant LunchS_d
$Lunch^+_{d,n} \in \{0,1\}$	$= 1$ ssi n termine la vacation d après LunchE_d
$Small_{d,n} \in \{0,1\}$	$= 1$ ssi la vacation d de n dure au plus SmallD
$\forall t \in \mathcal{T}, IsFi_{d,n,t} \in \{0,1\}$	$= 1$ ssi n commence la vacation d avec la tâche t
$\forall t \in \mathcal{T}, IsLa_{d,n,t} \in \{0,1\}$	$= 1$ ssi n termine la vacation d avec la tâche t
$\forall i \in \{1,2\}, Order_{d,n,i} \in \{0,1\}$	$= 1$ ssi la vacation d de l'employé n précede son repos i
$\forall \boldsymbol{n \in \mathcal{N}},$	**Pour tout employé n**
$\forall i \in \{1,2\}, Break_{i,n} \in \mathbb{Z}$	Début du $i^{\text{ème}}$ repos hebdomadaire de n
$\forall d \in [\![1;8]\!], Off_{d,n} \in \{0,1\}$	$= 1$ ssi n est en repos le jour d

TABLE 4.1 – Variables du modèle $\text{PLNE}_{|\text{U}=0|\Delta}$ pour SDPTSP|U = 0|Δ

Les variables $Break_{1,n}$ et $Break_{2,n}$ donnent respectivement les dates de début du premier et du second repos hebdomadaire pour chaque employé. Il est nécessaire de positionner deux repos hebdomadaires pour une seule semaine de manière à gérer le cas limite de cette contrainte glissante. En effet, si le repos hebdomadaire de la semaine précédente est positionné au début de la semaine précédente, alors le repos de la semaine courante doit être positionné au début de la semaine courante. Dans ce cas, le prochain repos hebdomadaire doit être positionné au plus tard au début de la semaine suivante, ce qui peut entrer en conflit avec certaines affectations de la semaine courante. Rappelons en effet que les tâches commençant le lundi matin avant 6h sont gérées en tant que tâches de nuit du dimanche. Autrement dit, il est nécessaire de positionner deux repos hebdomadaires pour une seule semaine, mais dans le cas général, le second repos sera positionné sur la semaine suivante et ne limitera pas les affectations de la semaine courante.

Les vacations des employés sont positionnées par rapport aux dates de début et de fin au plus tôt de ces repos hebdomadaires au moyen des variables binaires $Order_{d,n,i}$. Plus précisément, ces variables prennent la valeur 1 si la vacation d de l'employé n se termine avant son repos i et 0 si elle commence après. Les variables $Empty_{d,n}$ et $Off_{d,n}$ permettent de comptabiliser les vacations non travaillées et les jours de repos des employés. Cette distinction vient du fait qu'une vacation non travaillée ne correspond à un jour de repos que si la vacation précédente se termine suffisamment tôt. De plus, les vacations non travaillées doivent être identifiées de manière à calculer leur dates de début et de fin de manière spécifique.

Les variables $Lunch^-_{d,n}$ (respectivement $Lunch^+_{d,n}$) prennent la valeur 1 si et seulement la vacation d de l'employé n commence avant (respectivement termine après) la fenêtre horaire de la pause déjeuner. Les variables $Small_{d,n}$ prennent la valeur 1 si et seulement la vacation d de l'employé n dure au plus SmallD. Les variables $Lunch^-_{d,n}$, $Lunch^+_{d,n}$ et $Small_{d,n}$ interviennent dans le calcul de la pause déjeuner. Pour finir les variables $IsFi_{d,n,t}$ (respectivement $IsLa_{d,n,t}$) prennent la valeur 1 si et seulement si la tâche t est la première (respectivement la dernière) de la vacation d de l'employé n.

Contraintes métier du modèle PLNE$_{|\mathbf{U}=0|\Delta}$

Nous expliquons ici les contraintes du modèle PLNE$_{|\mathbf{U}=0|\Delta}$ en précisant à chaque fois les contraintes métier associées (C 1 à C 14 *cf.* Section 4.1.2). Tout d'abord, (4.3) et (4.4) permettent de calculer les écarts à la charge médicale idéale des employés ainsi que l'objectif du problème, *i.e.* l'inéquité entre les employés. La contrainte C 1 est vérifiée de manière indirecte via le calcul des dates de début et de fin des vacations. Ces dates sont en effet obtenues à partir des tâches affectées aux employés et toutes les tâches d'une même journée de travail appartiennent à la même vacation. Autrement dit, des tâches suffisamment espacées pour être associées à deux vacations différentes tout en respectant le repos minimum, mais suffisamment proches pour appartenir à la même journée de travail, conduiraient alors à un dépassement du temps présentiel maximal, car elles seraient associées à une seule vacation.

Le respect des tâches administratives obligatoires (C 2) ainsi que la prise en compte des compétences et des disponibilités (C 3) sont assurés par (4.5) et (4.6). Les contraintes (4.7) vérifient que les employés ne sont pas affectés à deux tâches concomitantes (C 4) alors que la contrainte C 6 est automatiquement respectée par construction du modèle, via l'intégrité des variables $Assign_{t,n}$. De manière similaire à C 1, la contrainte C 7 est vérifiée de manière indirecte via le calcul des dates de début et de fin des vacations. En effet, des tâches suffisamment proches pour être associées à une seule vacation sans dépasser le temps présentiel maximal, mais suffisamment éloignées pour appartenir à des journées de travail différentes, conduiraient alors à une violation de la contrainte de repos, car elles seraient associées à deux vacations différentes. Les contraintes (4.8) assurent que chaque tâche est assignée à exactement un employé. Nous verrons à la section suivante comment adapter cette contrainte de manière à être en mesure de construire des solutions avec une affectation partielle.

La contrainte C 8 est vérifiée de manière indirecte par cohérence entre les variables $Work_{d,n}$, $Lunch_{d,n}$, $First_{d,n}$ et $Last_{d,n}$. Les contraintes (4.9) et (4.10) permettent de borner le temps travaillé quotidiennement et hebdomadairement par les employés (C 9 & C 10). Les temps de repos minimum quotidien (C 11) et hebdomadaire (C 12) sont respectivement vérifiés grâce à (4.11) et (4.12) à (4.16). Plus précisément, (4.12) à (4.13) permettent de positionner un repos hebdomadaire pour chaque employé. Les journées de travail des employés, identifiées via leurs bornes $First_{d,n}$ et $Last_{d,n}$ peuvent soit finir avant le début de ce repos ($Order_{d,n,1} = 1$), soit commencer après la fin de ce repos ($Order_{d,n,1} = 0$), ce qui interdit tout recoupement entre le repos hebdomadaire et les journées de travail. Rappelons qu'il est nécessaire de positionner deux repos hebdomadaires sur une même semaine de manière à gérer le cas limite de la contrainte C 12. Le deuxième repos hebdomadaire est sujet aux mêmes contraintes que le premier (*cf.* (4.14) à (4.15)). Une contrainte additionnelle permet de gérer la distance maximale entre ces deux repos (4.16). La contrainte C 13 étant glissante, il est nécessaire de la vérifier entre les semaines précédente et courante ainsi qu'entre les semaines courante et suivante. Pour cela, nous utilisons (4.17) et (4.18). Pour finir, les contraintes (4.19) à (4.22) définissent les vacations associées à une pause déjeuner (C 14).

Contraintes de liaisons du modèle PLNE$_{|\mathbf{U}=0|\Delta}$

Les contraintes (4.23) à (4.28) assurent la cohérence entre les vacations des employés et les tâches qui leur sont affectées. Plus précisément, les contraintes (4.23) et (4.24) donnent respectivement les bornes supérieures et inférieures des variables $First_{d,n}$. Les dates de début des journées ne comprenant aucune tâche ($Empty_{d,n} = 1$) prennent la valeur ω (une constante représentant la fin de la semaine). De manière similaire, les contraintes (4.25) et (4.26) donnent respectivement les bornes inférieures et supérieures des variables $Last_{d,n}$. Les dates de fin des journées ne comprenant aucune

$$\min \Delta$$

$$s.t. \quad \forall (n_0, n_1) \in \mathcal{N}^2 \,|\, n_0 \neq n_1, \quad \Delta \geq \Delta_{n_0} - \Delta_{n_1} \tag{4.3}$$

$$\forall n \in \mathcal{N}, \quad \Delta_n = \sum_{t \in \mathcal{T}} proc_t \times Assign_{t,n} - target_n \tag{4.4}$$

Contraintes organisationnelles

$$\forall n \in \mathcal{N}, \forall t \in \mathcal{M}_n, \quad Assign_{t,n} = 1 \tag{4.5}$$

$$\forall n \in \mathcal{N}, \forall t \notin \mathcal{T}_n, \quad Assign_{t,n} = 0 \tag{4.6}$$

$$\forall n \in \mathcal{N}, \forall \mathcal{K} \in \mathcal{C}, \quad \sum_{t \in \mathcal{K}} Assign_{t,n} \leq 1 \tag{4.7}$$

$$\forall t \in \mathcal{E}, \quad \sum_{n \in \mathcal{N}} Assign_{t,n} = 1 \tag{4.8}$$

Contraintes légales ($\forall n \in \mathcal{N}$)

Temps travaillé maximum quotidien/hebdomadaire

$$\forall d \in \mathcal{D}, \quad Work_{d,n} \leq \textsc{WorkD}_{\textsc{max}} \tag{4.9}$$

$$\sum_{d \in \mathcal{D}} \quad Work_{d,n} \leq \textsc{WorkW}_{\textsc{max}} \tag{4.10}$$

Repos minimum quotidien/hebdomadaire

$$\forall d \in [\![1;6]\!], \quad First_{d+1,n} - Last_{d,n} \geq \textsc{BreakD}_{\textsc{min}} \tag{4.11}$$

$$\forall d \in \mathcal{D}, \quad Last_{d,n} - Break_{1,n} \leq (1 - Order_{d,n,1}) \times \omega \tag{4.12}$$

$$\forall d \in \mathcal{D}, \quad Break_{1,n} + \textsc{BreakW}_{\textsc{min}} - First_{d,n} \leq Order_{d,n,1} \times \omega \tag{4.13}$$

$$\forall d \in \mathcal{D}, \quad Last_{d,n} - Break_{2,n} \leq (1 - Order_{d,n,2}) \times \omega \tag{4.14}$$

$$\forall d \in \mathcal{D}, \quad Break_{2,n} + \textsc{BreakW}_{\textsc{min}} - First_{d,n} \leq Order_{d,n,2} \times \omega \tag{4.15}$$

$$Break_{2,n} - Break_{1,n} \leq \textsc{Week} \tag{4.16}$$

Jour de repos hebdomadaire

$$\sum_{d \in [\![2;8]\!]} Off_{d,n} \geq 1 \tag{4.17}$$

$$\sum_{d \in [\![1;7-off_n^{-1}]\!]} Off_{d,n} \geq 1 \tag{4.18}$$

Pause déjeuner

$$\forall d \in \mathcal{D}, \quad Lunch_{d,n} \leq Lunch^-_{d,n} \tag{4.19}$$

$$\forall d \in \mathcal{D}, \quad Lunch_{d,n} \leq Lunch^+_{d,n} \tag{4.20}$$

$$\forall d \in \mathcal{D}, \quad Lunch_{d,n} \leq (1 - Small_{d,n}) \tag{4.21}$$

$$\forall d \in \mathcal{D}, \quad Lunch_{d,n} \geq Lunch^-_{d,n} + Lunch^+_{d,n} - Small_{d,n} - 1 \tag{4.22}$$

Contraintes de liaisons $(\forall \boldsymbol{d} \in \boldsymbol{\mathcal{D}}, \forall \boldsymbol{n} \in \boldsymbol{\mathcal{N}})$

Nature et Positionnement des vacations

$$\forall t \in \mathcal{E}_{d,n}, \quad First_{d,n} \leq start_t \times Assign_{t,n} + (1 - Assign_{t,n}) \times \omega \tag{4.23}$$

$$First_{d,n} \geq Empty_{d,n} \times \omega + \sum_{t \in \mathcal{E}_{d,n}} start_t \times IsFi_{d,n,t} \tag{4.24}$$

$$\forall t \in \mathcal{E}_{d,n}, \quad Last_{d,n} \geq end_t \times Assign_{t,n} + (1 - Assign_{t,n}) \times \alpha \tag{4.25}$$

$$Last_{d,n} \leq Empty_{d,n} \times \alpha + \sum_{\forall t \in \mathcal{E}_{d,n}} end_t \times IsLa_{d,n,t} \tag{4.26}$$

$$\forall t \in \mathcal{E}_{d,n}, \quad IsFi_{d,n,t} \leq Assign_{t,n} \quad \textbf{et} \quad IsLa_{d,n,t} \leq Assign_{t,n} \tag{4.27}$$

$$\sum_{t \in \mathcal{E}_{d,n}} IsFi_{d,n,t} = (1 - Empty_{d,n}) \quad \textbf{et} \quad \sum_{t \in \mathcal{E}_{d,n}} IsLa_{d,n,t} = (1 - Empty_{d,n}) \tag{4.28}$$

$$\forall d \in [\![1; 8]\!], \quad \sum_{t \in \mathcal{O}_d} Assign_{t,n} \leq (1 - Off_{d,n}) \times \text{card}(\mathcal{O}_d) \tag{4.29}$$

Pause déjeuner

$$\forall t \in \mathcal{L}_d^- \quad Lunch_{d,n}^- \geq Assign_{t,n} \quad \textbf{et} \quad \forall t \in \mathcal{L}_d^+ \quad Lunch_{d,n}^+ \geq Assign_{t,n} \tag{4.30}$$

$$Lunch_{d,n}^- \leq \sum_{t \in \mathcal{L}_d^-} Assign_{t,n} \quad \textbf{et} \quad Lunch_{d,n}^+ \leq \sum_{t \in \mathcal{L}_d^+} Assign_{t,n} \tag{4.31}$$

$$Last_{d,n} - First_{d,n} + (\omega - \alpha + \textsc{SmallD} + 1) \times Small_{d,n} \geq \textsc{SmallD} + 1 \tag{4.32}$$

$$Last_{d,n} - First_{d,n} \leq (\textsc{Span}_{\textsc{max}} - \textsc{SmallD}) \times (1 - Small_{d,n}) + \textsc{SmallD} \tag{4.33}$$

Temps travaillé

$$Work_{d,n} \leq \textsc{WorkD}_{\textsc{max}} \times (1 - Empty_{d,n}) \tag{4.34}$$

$$Work_{d,n} \leq Last_{d,n} - First_{d,n} - \textsc{LunchP} \times Lunch_{d,n} + (\omega - \alpha) \times Empty_{d,n} \tag{4.35}$$

$$Work_{d,n} \geq Last_{d,n} - First_{d,n} - \textsc{LunchP} \times Lunch_{d,n} \tag{4.36}$$

$$Work_{d,n} \geq \sum_{t \in \mathcal{E}_{d,n}} Assign_{t,n} \times proc_t \tag{4.37}$$

Initialisation $(\forall \boldsymbol{n} \in \boldsymbol{\mathcal{N}})$

$$\text{si } work_n^{-1} \geq \textsc{DayS}_1, \text{alors} \quad Off_{1,n} = 0 \tag{4.38}$$

$$First_{1,n} \geq \textsc{BreakD}_{\textsc{min}} - work_n^{-1} \tag{4.39}$$

$$Break_{1,n} \geq -work_n^{-1} \tag{4.40}$$

$$Break_{1,n} \geq -break_n^{-1} - \textsc{BreakW}_{\textsc{min}} \quad \textbf{et} \quad Break_{2,n} \geq \textsc{DayS}_2 \tag{4.41}$$

tâche ($Empty_{d,n} = 1$) prennent la valeur α (une constante représentant le début de la semaine).

Les contraintes (4.27) à (4.28) assurent la cohérence des variables $IsFi_{d,n,t}$ et $IsLa_{d,n,t}$ par rapport aux tâches affectées aux employés. Plus précisément, ces contraintes permettent de vérifier que les variables $First_{d,n}$ (respectivement $Last_{d,n}$) correspondent à la plus petite (respectivement la plus grande) date de début (respectivement date de fin) parmi les tâches affectées à l'employé n durant la vacation d. De cette manière, la pause déjeuner ne peut être positionnée aux extrémités des vacations. Les contraintes (4.29) vérifient que seuls les horaires de travail entièrement non travaillés sont comptabilisés en tant que jours de repos. Cela s'exprime par le fait qu'aucune des tâches concomitantes à un tel intervalle n'est affectée à l'employé considéré.

Les contraintes (4.30) à (4.31) permettent d'identifier les vacations commençant avant le début de la fenêtre horaire de la pause déjeuner et se terminant après la fin de la fenêtre horaire de la pause déjeuner. Les contraintes (4.32) à (4.33) permettent d'identifier les vacations dont le temps présentiel est strictement supérieur à SmallD. Les contraintes (4.34) à (4.37) permettent d'obtenir le temps de travail correspondant aux vacations. Ces contraintes prennent en compte les différents cas possibles, *i.e.* vacations vides, vacations sans pause déjeuner et vacation avec pause déjeuner. Remarquons que les valeurs α et ω ont été choisies de manière à ce que les autres contraintes restent cohérentes pour les vacations vides.

Les contraintes (4.38) à (4.41) permettent de prendre en compte la semaine passée, de manière à ce que les contraintes glissantes soient respectées en début de semaine. Plus précisément, les contraintes (4.38) stipulent que les employés dont le dernier travail de la semaine précédente s'est terminé après DayS_1 (lundi de la semaine courante à 6h du matin), ne peuvent être en repos le lundi. Les contraintes (4.39) vérifient le repos journalier entre la semaine passée et la semaine courante. Les contraintes (4.40) permettent de gérer le cas limite des contraintes (4.11) et (4.12) : le repos hebdomadaire de la semaine courante débute après le dernier travail de la semaine passée, et le premier travail du lundi débute après le repos journalier consécutif au dernier travail de la semaine passée. Pour finir, les contraintes (4.41) permettent d'initialiser les dates de début au plus tôt des repos hebdomadaires.

4.3.2 Modélisation PLNE autour de SDPTSP||U

Le modèle que nous avons présenté à la section précédente permet de minimiser l'inéquité des affectations complètes. S'il n'existe aucune solution complète, le modèle $\text{PLNE}_{|\text{U}=0|\Delta}$ est alors en mesure de prouver l'infaisabilité de l'instance, mais il ne permet pas de la quantifier et cela peut prendre beaucoup de temps. Pour cette raison, nous nous intéressons à présent au problème SDPTSP||U qui vise à minimiser le nombre de tâches non affectées. Puisque nous cherchons ici à statuer sur la réalisabilité des instances de manière à être en mesure d'évaluer plus finement les résultats des autres méthodes, nous ne prenons pas en compte l'équité.

$\text{PLNE}_{||\text{U}}$, un PLNE pour SDPTSP||U

À partir du modèle $\text{PLNE}_{|\text{U}=0|\Delta}$, nous pouvons facilement obtenir un deuxième modèle permettant de résoudre le SDPTSP||U. En effet, si l'on cherche uniquement à trouver une affectation des tâches aux employés, il n'est alors pas nécessaire de calculer l'inéquité entre les employés, ce qui permet de supprimer les variables Δ et Δ_n, ainsi que les contraintes (4.3) à (4.4). En revanche, il devient nécessaire de calculer le nombre de tâches non affectées, ce qui se fait au moyen de la contrainte (4.42). Nous relaxons également les contraintes (4.8) en vérifiant simplement que chaque tâche est affectée à au plus un employé (4.43). Ces modifications nous conduisent à un nouveau modèle, noté $\text{PLNE}_{||\text{U}}$.

$$\mathrm{U} = |\mathcal{E}| - \sum_{n\in\mathcal{N},t\in\mathcal{E}} Assign_{t,n} \tag{4.42}$$

$$\forall t \in \mathcal{E}, \quad \sum_{n\in\mathcal{N}} Assign_{t,n} \leq 1 \tag{4.43}$$

PLNE pour le SDPTSP sans pause déjeuner

Le modèle que nous avons présenté à la section précédente permet de minimiser le nombre de tâches non affectées, sans prendre en compte l'équité des solutions. Malgré cette première simplification, des résultats préliminaires ont montré que ce modèle n'arrivait ni à résoudre les instances de moyenne et grande taille ($\mathrm{I_t}$= 200, 300 et 400), ni même à calculer de borne inférieure pour U. L'objectif principal étant d'évaluer l'existence de solutions complètes, nous avons cherché à simplifier le modèle $\mathrm{PLNE}_{||\mathrm{U}}$. Pour cela nous avons retiré la contrainte portant sur la pause déjeuner, car cela entraîne une simplification importante du modèle et permet malgré tout d'obtenir une bonne borne inférieure pour U. La contrainte de la pause déjeuner est en effet l'une des contraintes les plus faibles du modèle $\mathrm{PLNE}_{||\mathrm{U}}$, mais elle requiert un nombre important de variables et de contraintes. En supprimant cette contrainte, nous obtenons un modèle plus léger, sans modifier fortement la réalisabilité des instances. Bien entendu, une instance admettant une solution complète avec ce nouveau modèle, noté $\mathrm{PLNE}_{||\mathrm{U}_r}$, peut ne pas admettre de solution complète avec $\mathrm{PLNE}_{||\mathrm{U}}$. Par conséquent, nous utiliserons le modèle $\mathrm{PLNE}_{||\mathrm{U}_r}$ uniquement pour prouver l'infaisabilité de certaines instances.

Le modèle $\mathrm{PLNE}_{||\mathrm{U}_r}$ s'obtient à partir du modèle $\mathrm{PLNE}_{||\mathrm{U}}$, en retirant les variables et les contraintes portant sur la pauses déjeuner, ce qui permet une simplification importante du modèle. Tout d'abord, les variables $Lunch_{d,n}$, $Lunch^{-}_{d,n}$, $Lunch^{+}_{d,n}$ et $Small_{d,n}$ deviennent inutiles puisqu'elles ne servent qu'à la pause déjeuner. D'autre part, les variables $IsFi_{d,n}$ et $IsLa_{d,n}$, qui interviennent dans le calcul des dates de début et de fin des vacations, peuvent également être retirées du modèle. Ainsi, au lieu de fixer précisément les vacations des employés au début de la première tâche et à la fin de la dernière tâche, nous pouvons simplement garantir que les vacations commencent avant le début de la première tâche et se terminent après la fin de la dernière tâche. Cela revient en fait à poster les contraintes légales sur « l'enveloppe des vacations » au lieu de les poster sur le « noyau des vacations ». Cette modification est envisageable car il n'est plus nécessaire de vérifier que la pause déjeuner n'est pas positionnée aux extrémités des vacations.

La suppression des variables devenues superflues conduit tout d'abord à supprimer les contraintes métier (4.19) à (4.22) ainsi que les contraintes de liaison (4.30) à (4.33) qui portent toutes sur la pause déjeuner. D'autre part, les contraintes de liaison portant sur le temps travaillé peuvent également être simplifiées, puisqu'il n'est plus utile de faire de distinction entre le temps de présence et le temps de présence travaillé. Plus précisément, les contraintes (4.37) peuvent être supprimées, tandis que les contraintes (4.35) à (4.36) sont remplacées par les contraintes (4.44) à (4.45). Pour finir les contraintes de liaison (4.24), (4.26), (4.27) et (4.28) qui positionnent précisément les vacations des employés peuvent être supprimées.

$$Work_{d,n} \leq Last_{d,n} - First_{d,n} + (\omega - \alpha) \times Empty_{d,n} \tag{4.44}$$

$$Work_{d,n} \geq Last_{d,n} - First_{d,n} \tag{4.45}$$

Méthode utilisée	Temps de calcul	Instances infaisables
$H_{\underline{U}}$	-	41
$PLNE_{\|\|U}$	1 h	78
$PLNE_{\|\|U_r}$	1 h	98
$PLNE_{\|\|U_r}$	20 h	100
Total		102

TABLE 4.2 – Nombre d'instances infaisables pour SDPTSP|U=0|-

4.4 Résultats expérimentaux

4.4.1 Preuves d'infaisabilité pour SDPTSP|U=0|-

Nous avons utilisé $H_{\underline{U}}$ (*cf.* Algorithme 4.2) ainsi que les modèles $PLNE_{||U}$ et $PLNE_{||U_r}$[1] pour statuer sur la faisabilité des instances générées (*cf.* Section 2.3.2, p. 33), vis-à-vis de SDPTSP|U=0|. Les résultats obtenus, présentés à la Table 4.2, montrent clairement l'intérêt du modèle $PLNE_{||U_r}$ par rapport au modèle $PLNE_{||U}$ puisqu'il permet d'obtenir 20 preuves supplémentaires d'infaisabilité au bout d'une heure de calcul. En revanche, $PLNE_{||U}$ permet d'obtenir 287 solutions complètes au bout d'une heure, prouvant ainsi la faisabilité de ces instances, alors que le modèle $PLNE_{||U_r}$ ne peut pas statuer sur la faisabilité des instances. Toutes méthodes confondues, nous obtenons ainsi des preuves d'infaisabilité (respectivement de faisabilité) pour 102 (respectivement 287) instances sur un total de 720, ce qui laisse 331 instances ouvertes. Comme cela est souligné par les Tables 4.3 et 4.4, plus l'instance est chargée ($I_v = 1\,000$) et petite ($I_t = 100$), plus il est probable qu'elle soit infaisable. À l'inverse, le paramètre I_c, définissant les compétences prises en compte, semble avoir un impact très limité sur la réalisabilité des instances. Par conséquent, nous évaluerons systématiquement les résultats selon les paramètres I_t et I_v qui semblent les plus déterminants. Notons que l'augmentation du temps de calcul n'est visiblement pas un bon moyen de fermer les instances restantes, comme le montrent les résultats obtenus par $PLNE_{||U_r}$ au bout de 20 heures de résolution.

I_t	I_v			
	600	800	1 000	Total
100	6/60	13/60	39/60	58/180
200	0/60	5/60	25/60	30/180
300	2/60	2/60	4/60	8/180
400	1/60	1/60	4/60	6/180
Total	9/240	21/240	72/240	102/720

TABLE 4.3 – Répartition des instances infaisables selon I_t et I_v.

I_t	I_v			
	600	800	1 000	Total
I_{cc}	3/120	10/120	39/120	52/360
I_{cr}	6/120	11/120	33/120	50/360
Total	9/240	21/240	72/240	102/720

TABLE 4.4 – Répartition des instances infaisables selon I_c et I_v.

4.4.2 Intérêt de la descente locale H_Δ

Afin d'évaluer l'efficacité de la descente locale H_Δ, nous avons comparé les valeurs de l'inéquité obtenues par les modèles $PLNE_{|U=0|\Delta}$ et $PLNE_{||U}$. Pour chaque modèle, nous donnons l'inéquité moyenne (en minutes) avant et après l'utilisation de H_Δ. Les résultats obtenus, présentés à la Table 4.5, illustrent clairement l'intérêt de H_Δ qui permet de réduire drastiquement l'inéquité sur les solutions complètes du modèle $PLNE_{||U}$. De plus, même sur les solutions du modèle $PLNE_{|U=0|\Delta}$, H_Δ reste très

[1] Les algorithmes sont implémentés en JAVA et nous utilisons *Cplex 12.4* (configuration par défaut) [114] pour implémenter les modèles PLNE. Les tests sont réalisés sur un Intel Core i3-540 (3.06 GHz & 8G RAM)

Modèle	Solutions complètes	Δ avant H_Δ	Δ après H_Δ
$PLNE_{\|\|U}$	205	1195	262
$PLNE_{\|U=0\|\Delta}$	86	62	44

TABLE 4.5 – Valeur moyenne de Δ après 10 min

efficace. Puisque cette descente locale semble efficace pour réduire Δ, nous l'utiliserons par la suite au sein des différentes méthodes de résolution développées.

4.5 Conclusion

Dans ce chapitre, nous avons formalisé les différentes variantes du SDPTSP que nous étudions dans cette thèse. Nous avons également donné des algorithmes de pré-traitement et de post-traitement, ainsi qu'une méthode permettant de calculer une borne inférieure sur le nombre de tâches non affectées. Nous avons également proposé plusieurs variantes d'un même programme linéaire en nombres entiers. Ces différentes variantes nous donnent des éléments d'analyses intéressants pour la suite de notre étude. En ce qui concerne la réalisabilité des instances, nous avons montré qu'une centaine d'entre elles n'admettent pas de solutions complètes. Ces instances sont intéressantes car elles correspondent au cas où Biotrial doit embaucher des intérimaires pour couvrir toutes les tâches. En ce qui concerne l'obtention de solutions réalisables, les modèles proposés conduisent à des résultats très mitigés. Après 10 minutes de calcul, les meilleurs résultats sont obtenus par le modèle $PLNE_{||U}$ qui obtient une solution complète pour environ un tiers des instances a priori réalisables. Le modèle $PLNE_{|U=0|\Delta}$, quant à lui, trouve très peu de solutions complètes. Puisque les décideurs de Biotrial souhaitent obtenir une bonne solution en moins de 5 minutes, il semble essentiel de concevoir des méthodes dédiées. Dans la suite de cette thèse, nous nous concentrerons sur le problème SDPTSP||U,Δ qui correspond aux attentes de Biotrial.

5

Une méthode en deux phases

Nous présentons ici une méthode en deux phases (M2P) pour le SDPTSP||U,Δ. Ce type d'approche est particulièrement efficace et naturel pour les problèmes portant sur plusieurs niveaux de décision, ce qui est le cas ici. La méthode que nous proposons consiste à fixer les horaires du personnel (Phase 1) puis à affecter les tâches aux employés (Phase 2). Nous itérons entre ces deux phases de manière à faire évoluer les horaires du personnel selon l'affectation courante. Nous proposons pour cela plusieurs opérateurs de voisinage permettant de modifier les vacations des employés. Nous justifions l'architecture globale de la méthode, ainsi que son paramétrage via une série de tests expérimentaux, et nous concluons sur l'intérêt de cette méthode.

Sommaire

5.1 Utilisations des méthodes en deux phases

Bien que les méthodes en deux phases ne garantissent pas l'optimalité des solutions, elles présentent généralement un bon compromis entre le temps de calcul et la qualité des solutions, du fait de la décomposition en sous-problèmes. Ces méthodes sont par conséquent très pertinentes lorsqu'il s'agit de trouver rapidement une bonne solution. Il est également très naturel de recourir à ce genre de méthode lorsqu'il s'agit d'aborder un problème portant sur plusieurs niveaux de décision. Les problèmes de planification de personnel, par exemple, sont tout à fait propices à l'utilisation de méthodes en deux phases.

Brucker *et al.* [35] ainsi que Valouxis *et al.* [188] proposent une méthode en deux phases pour un problème de planification d'infirmières. Dans une première phase, Brucker *et al.* établissent une liste de plannings individuels pour chaque employé, puis, dans une seconde phase, ils construisent le planning global en sélectionnant un planning individuel pour chaque employé. Cette seconde phase est répétée plusieurs fois en modifiant l'ordre dans lequel les plannings individuels sont sélectionnés de manière à trouver des solutions améliorantes. Autrement dit, il s'agit de générer un voisinage sur les plannings potentiels (phase 1) puis de l'explorer (phase 2). À l'inverse, la méthode proposée par Valouxis *et al.* est plus itérative. La première phase consiste à déterminer les jours de travail des infirmières grâce à une succession de recherches locales appliquées sur une solution initiale obtenue de manière aléatoire. La seconde phase consiste quant à elle à préciser les horaires de travail réalisés par chaque infirmière. Ces deux étapes sont répétées jusqu'à épuisement du temps imparti. La diversité des solutions obtenues en phase 1 est assurée par le caractère aléatoire de la solution initiale.

Toujours dans le domaine de la planification de personnel, Lequy *et al.* [138] proposent une méthode en deux phases pour un problème d'affectation d'activités et de tâches. Il s'agit d'affecter des tâches non préemptables et des activités préemptables à des employés compétents de manière à respecter les fenêtres horaires de réalisation des tâches tout en minimisant une somme de pénalités comprenant notamment le coût de non affectation d'une tâche. La première phase consiste à affecter les tâches alors que la seconde phase consiste à affecter les activités. Les auteurs proposent quatre stratégies différentes permettant de résoudre la seconde phase. Ces quatre stratégies correspondent à quatre niveaux de flexibilité vis-à-vis de la remise en cause de la solution de la phase 1 : aucune flexibilité, flexibilité vis-à-vis du choix des employés affectés aux tâches, flexibilité vis-à-vis de la date de début des tâches, ou bien une combinaison des deux dernières flexibilités. Les auteurs montrent expérimentalement que la stratégie offrant le plus de flexibilité conduit systématiquement à de meilleurs résultats.

Puisque le SDPTSP||U,Δ porte à la fois sur la planification des horaires du personnel et sur l'affectation des tâches, il semble naturel de le décomposer en un problème de planification de personnel, suivi d'un problème d'affectation de tâches, ce qui s'apparente à la décomposition proposée par Valouxis *et al.* [188]. La différence principale entre notre approche et celle de Valouxis *et al.* est que nous proposons de faire évoluer les horaires du personnel d'une itération à l'autre via une succession de modifications, alors que Valouxis *et al.* génèrent à chaque itération une nouvelle solution aléatoire.

5.2 Structure de M2P

5.2.1 Fonctionnement général de M2P

L'idée générale de M2P consiste à décomposer le SDPTSP||U,Δ en deux sous-problèmes résolus successivement et de manière itérative. La première phase fixe les horaires du personnel, alors que la seconde phase affecte les tâches aux employés en prenant en compte leurs horaires de travail. D'une

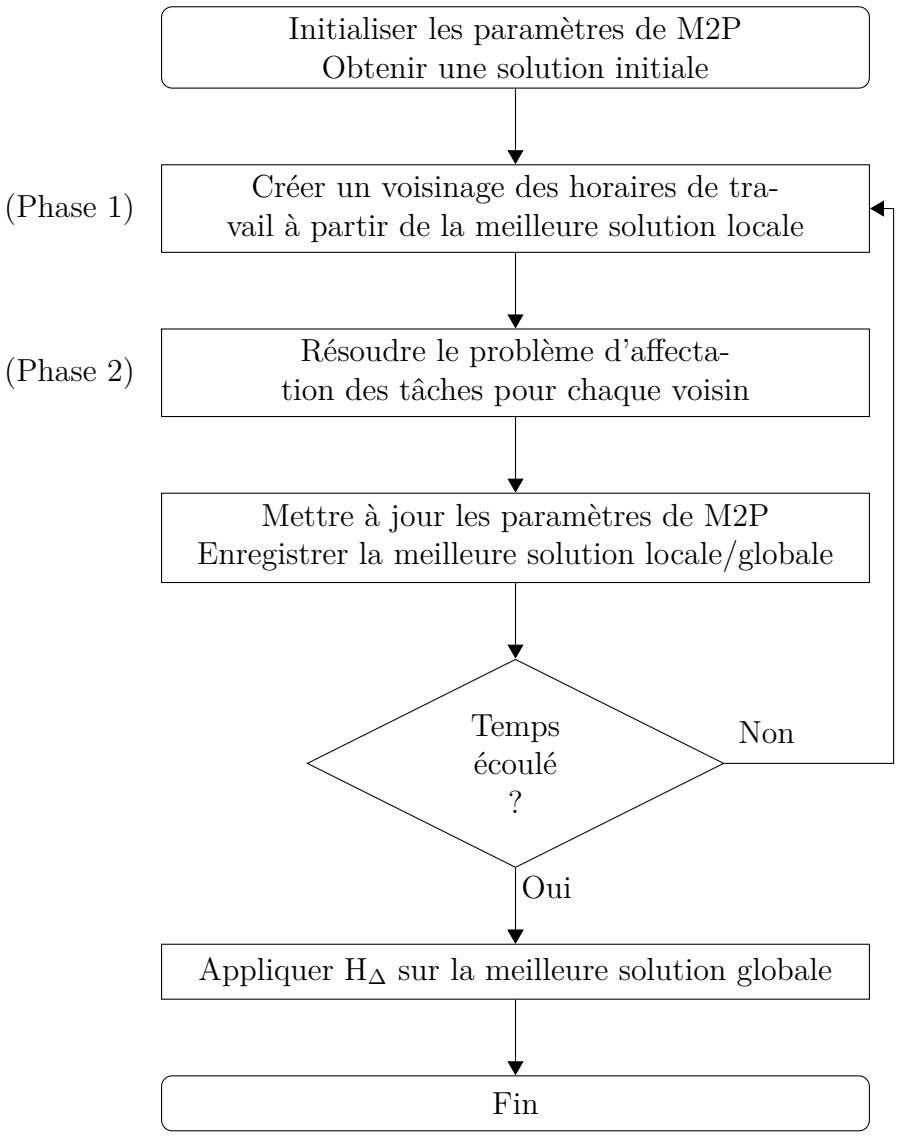


FIGURE 5.1 – Logigramme simplifié de M2P.

itération à l'autre, la phase 1 permet de modifier les horaires du personnel selon les besoins de la phase 2. Pour cela, nous utilisons plusieurs opérateurs de voisinage prenant en entrée les horaires du personnel ainsi que l'affectation courante des tâches et donnant en sortie des horaires modifiés. La descente locale H_Δ est utilisée sur la meilleure solution pour réduire l'inéquité (*cf.* Section 4.2.3, p. 66). La structure générale de M2P est illustrée par la Figure 5.1.

5.2.2 Phase 1 : construction des horaires de travail du personnel

Nous nous intéressons ici à la première phase de M2P. À ce stade de la résolution, nous disposons d'une solution courante représentée par l'affectation des tâches mais également par les horaires de travail du personnel. Compte tenu des éventuelles tâches non affectées, ainsi que des éventuels déséquilibres de charge des employés, il s'agit alors de modifier les horaires actuels de manière à favoriser l'obtention d'une meilleure solution lors de la résolution de la seconde phase. Puisque cette première phase porte uniquement sur la construction des horaires du personnel, les contraintes relatives à l'affectation des tâches ne sont pas prises en compte.

Opérateur	Fonctionnement
Ajouter(ρ)	Choisir un employé parmi un pourcentage ρ des employés les plus en dessous de leur charge idéale et lui ajouter une journée de travail. Seuls les employés ayant moins de 5 journées de travail sont éligibles. Les journées de travail sont rajoutées aléatoirement pour couvrir une des tâches du planning.
Étendre(ρ)	Choisir un employé parmi un pourcentage ρ des employés les plus en dessous de leur charge idéale, choisir l'une de ses journées de travail et l'étendre au maximum soit à son début soit à sa fin. Les contraintes légales doivent être respectées.
Retirer(ρ)	Choisir un employé parmi un pourcentage ρ des employés les plus au-dessus de leur charge idéale, choisir l'une de ses journées de travail et la retirer.
Déplacer	Choisir un horaire de travail et le déplacer d'une durée aléatoire variant entre 4 et 8 heures.
Équilibrer	Choisir l'un des employés le plus au-dessus de sa charge idéale et choisir l'une de ses tâches. Parmi les autres employés auxquels il est possible d'ajouter une journée de travail couvrant cette tâche, choisir l'un d'entre eux, et ajouter cette journée de travail.
Couvrir	S'il existe des tâches non affectées, choisir l'une d'entre elles, sinon, choisir l'une de celles qui est la moins couverte par les horaires de travail actuels. Parmi les journées de travail pouvant être étendues de manière à couvrir cette tâche, choisir l'une d'entre elles et l'étendre pour couvrir la tâche sélectionnée.

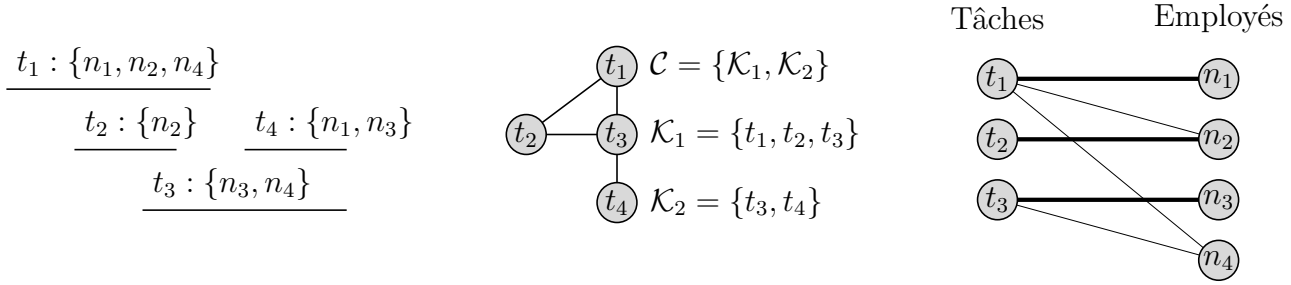
TABLE 5.1 – Opérateurs de voisinage sur les horaires du personnel

Nous proposons cinq opérateurs de voisinage *Ajouter*(ρ), *Étendre*(ρ), *Retirer*(ρ), *Déplacer*, *Équilibrer* et *Couvrir* (*cf.* Table 5.1). Chacun de ces opérateurs permet d'obtenir un seul voisin des horaires courants. Il est par conséquent nécessaire d'utiliser plusieurs fois le même opérateur, ou bien plusieurs opérateurs différents, de manière à obtenir un voisinage sur les horaires du personnel. Afin de diversifier les voisins explorés, les opérateurs proposés effectuent de nombreux choix aléatoires. A priori, l'opérateur *Couvrir* se focalise davantage sur l'obtention d'une solution complète, alors que les opérateurs *Ajouter*(ρ), *Étendre*(ρ), *Retirer*(ρ) et *Équilibrer* se focalisent davantage sur l'obtention d'une solution équitable. L'opérateur *Déplacer* permet quant à lui de diversifier les horaires du personnel. Notons cependant que plus le paramètre ρ est proche de 1, moins les opérateurs *Ajouter*(ρ), *Étendre*(ρ) et *Retirer*(ρ) se focalisent sur l'équité et plus ils contribuent à diversifier les horaires du personnel.

5.2.3 Phase 2 : affectation des tâches à horaires de travail fixés

Nous nous intéressons ici au problème d'affectation des tâches, selon le critère d'équité défini à la Section 4.1.3 (p. 61). Les horaires de travail du personnel, obtenus en phase 1, constituent alors une donnée du problème. Il s'agit par conséquent de résoudre un Fixed Job Scheduling Problem, selon un critère d'équité. Les contraintes du problème se résument alors aux contraintes de compétence et de disponibilité ainsi qu'aux contraintes de non préemption des tâches, de non ubiquité des employés et de respect de la pause déjeuner. Les deux premières contraintes sont traitées simplement en prétraitement puisqu'elles portent directement sur les données du problème. Par ailleurs, puisque nous avons opté pour une modélisation sous la forme d'une affectation tâche-employé (*cf.* Section 4.1.4, p. 63), la contrainte de non préemption est automatiquement respectée. Nous nous focaliserons par conséquent sur les contraintes de non ubiquité des employés et de pause déjeuner.

La contrainte de non ubiquité des employés peut être prise en compte via l'ensemble $\mathcal{C}$ des cliques maximales de tâches concomitantes. En effet, les tâches $\mathcal{C}_\mathcal{K}$ d'une même clique $\mathcal{K} \in \mathcal{C}$ doivent être



(a) Exemple à 4 tâches et 4 employés. (b) Cliques maximales. (c) Couplage maximum pour $\mathcal{K}_1$.

FIGURE 5.2 – Exemple de graphe d'affectation relatif à une clique donnée.

affectées à des employés différents. L'ensemble $\mathcal{W}_\mathcal{K}$ des employés pouvant être affecté à au moins une tâche de $\mathcal{C}_\mathcal{K}$ étant connu, il est alors possible de construire le graphe d'affectation $G_\mathcal{K} = (\mathcal{C}_\mathcal{K} \cup \mathcal{W}_\mathcal{K}, \mathcal{E})$, où $\mathcal{E}$ représente l'ensemble des affectations possibles, *i.e.* respectant les contraintes de compétence et de disponibilité. La résolution du problème de couplage maximum au sein de $G_\mathcal{K}$ permet alors d'affecter un maximum de tâches de l'ensemble $\mathcal{C}_\mathcal{K}$. Puisque l'affectation ainsi obtenue ne tient pas compte de la pause déjeuner, il peut être nécessaire de retirer certaines affectations du couplage, pour que toutes les vacations respectent la contrainte C 14. Autrement dit, il est possible d'obtenir une solution en résolvant une succession de problèmes de couplage maximum dans les graphes d'affectation correspondants. Puisque les cliques sont fréquemment concomitantes, les décisions prises sur une clique impactent automatiquement les cliques voisines. Par conséquent, il est nécessaire de mettre à jour $G_\mathcal{K}$ avant de chercher son couplage maximum. Par exemple, à la Figure 5.2, la tâche t_3 appartient à la fois à $\mathcal{K}_1$ et $\mathcal{K}_2$. De plus, la tâche t_4 ne peut être affectée qu'aux employés n_1 et n_3. Par conséquent, suite à l'affectation de t_3 à n_3 au sein de la clique $\mathcal{K}_1$, la mise à jour de $G_{\mathcal{K}_2}$ requiert de supprimer n_3 et t_3 de $G_{\mathcal{K}_2}$, ce qui conduit à affecter t_4 à n_1.

La recherche d'une succession de couplages maximum présente l'inconvénient de raisonner de manière extrêmement locale. En effet, pour chaque couplage maximum, l'affectation courante n'est prise en compte qu'au travers des contraintes de non ubiquité et de pause déjeuner. Autrement dit, cela ne permet pas de prendre en compte la capacité de mobilisation des employés sur le reste de l'horizon. Pour raisonner de manière plus globale, nous proposons d'associer à chaque arête un indicateur heuristique correspondant a priori à l'intérêt de l'affectation correspondante. Ainsi, plus une affectation semble intéressante, plus le poids de l'arête correspondante est élevé. Au fil de la résolution, ces poids varient de manière à prendre en compte l'état courant de l'affectation. Il devient alors possible, par exemple, de favoriser les affectations recourant à des employés encore très en dessous de leur charge idéale, ou bien ceux qui ne peuvent plus être assignés qu'à un nombre très restreint de tâches. Ainsi, au lieu de résoudre une succession de couplages maximum, nous proposons de résoudre une succession de couplages de poids maximum. Pour cela, nous utilisons l'Algorithme Hongrois [131]. Bien entendu, le poids des arêtes peut être calculé selon différentes stratégies, et les cliques peuvent être explorées de différentes manières. Nous proposons par la suite plusieurs stratégies permettant de pondérer les arêtes et d'explorer les cliques. Ces stratégies, notées respectivement $\mathcal{S}p$ et $\mathcal{S}c$ sont récapitulées aux Tables 5.2 et 5.3. Le croisement de ces différentes stratégies conduit à un total de 45 variantes pour la phase 2, ce qui permet de diversifier les solutions obtenues. Le fonctionnement général de la seconde phase est détaillé par l'Algorithme 5.1.

En ce qui concerne les stratégies d'exploration des cliques ($\mathcal{S}c$), *Charge*, *Charge+Pot*, *3.Charge+Pot*, *10.Charge+Pot* et *20.Charge+Pot* permettent de distinguer les employés les uns des autres via le

Stratégie	$W(w,t)$	$W(w,t)$ est d'autant plus grand que w est...
Charge	$Ch(w)$	... en sous-charge.
Charge+Pot	$Ch(w) - Po(w)$	... en sous-charge et que son temps de travail potentiel est faible.
3.Charge+Pot	$3.Ch(w) - Po(w)$	
10.Charge+Pot	$10.Ch(w) - Po(w)$	
20.Charge+Pot	$20.Ch(w) - Po(w)$	
NCritiQ	$-nQ(w,t)$	... contraint dans ses affectations (en nombre).
DCritiQ	$-dQ(w,t)$	... contraint dans ses affectations (en temps).
Charge+NCritiQ	$Ch(w) - nQ(w,t)$	... en sous-charge et contraint dans ses affectations (en nombre).
Charge+DCritiQ	$Ch(w) - dQ(w,t)$	... en sous-charge et que la durée des affectations alternatives est faible.

$Ch(w)$ écart entre la charge idéale de w et sa charge courante.
$Po(w)$ somme des durées des tâches affectables à w.
$nQ(w,t)$ nombre de tâches concomitantes à t et affectables à w.
$dQ(w,t)$ durées cumulées des tâches concomitantes à t et affectables à w.

TABLE 5.2 – $\mathcal{S}p$: stratégies de calcul du poids $W(w,t)$ des arêtes (w un employé, t une tâche)

poids qui leur est associé. En revanche, pour un même employé, toutes les tâches sont a priori équivalentes. L'utilisation de coefficients dans la stratégie *Charge+Pot* permet de faire varier l'importance relative de la charge restante des employés par rapport au travail potentiel restant des employés. Tous les coefficients testés sont supérieurs à 1, car la charge restante des employés (*i.e.* l'écart à leur charge idéale) est a priori plus faible que le travail potentiel restant. A contrario, les stratégies *NCritiQ*, *DCritiQ*, *Charge+NCritiQ* et *Charge+DCritiQ* recourent à un raisonnement plus local permettant de distinguer chaque affectation. Ces stratégies reposent en partie sur la notion de *criticité* d'un employé vis-à-vis d'une tâche donnée. Plus précisément, l'affectation de la tâche t à l'employé w présente une criticité égale à l'opposé de $nQ(w,t)$. Par conséquent, plus un employé est compétent et disponible, plus $nQ(w,t)$ est grand, et donc, plus sa criticité est faible. Autrement dit, les employés ne pouvant réaliser qu'un petit nombre de tâches sont plus critiques que les autres. Notons que nous utilisons ici une criticité « locale », puisque nous calculons cette criticité par rapport à la fenêtre horaire d'une tâche. Nous nous inspirons ici de la notion de criticité originellement introduite par [25] de manière à évaluer l'utilité d'un employé par rapport à une charge de travail.

Nous présentons à présent les différentes stratégies de pondération des arêtes ($\mathcal{S}p$). Les stratégies *Chronologique* et *Durée* consistent respectivement à explorer les cliques par ordre chronologique et par ordre décroissant de durée des tâches. Afin de bénéficier pleinement du concept de criticité introduit précédemment, nous proposons également trois autres stratégies, *Densité*, *DensitéDurée* et *DuréeDensité*, qui reposent sur la notion de clique *dense*. Nous considérons qu'une clique est d'autant plus *dense* que le nombre de tâches de la clique est supérieur au nombre d'employés assignables à au moins l'une de ces tâches. Autrement dit, plus une clique est dense, plus il est probable que toutes les tâches ne puissent être affectées à un employé. Par conséquent, plus une clique est dense, plus il semble important d'évaluer finement la criticité des employés.

5.2.4 Obtention d'une solution initiale

Puisque la première phase de M2P requiert une solution en entrée pour construire un voisinage sur les horaires de travail, la toute première étape de M2P consiste à construire une solution initiale. Pour cela, nous proposons une méthode heuristique fonctionnant en deux temps. La méthode fixe tout d'abord les horaires du personnel pour couvrir au mieux le profil de charge. La seconde étape consiste

Stratégie	Choix de la clique à explorer
Chronologique	Sélectionner la première clique dans l'ordre chronologique.
Durée	Sélectionner la clique contenant la plus longue tâche, puis la première clique.
Densité	Sélectionner la clique présentant la plus petite différence entre le nombre d'employés disponibles et le nombre de tâches à affecter, puis la première clique.
DensitéDurée	Sélectionner la clique présentant la plus petite différence entre le nombre d'employés disponibles et le nombre de tâches à affecter, puis la clique contenant la plus longue tâche, puis la première clique.
DuréeDensité	Sélectionner la clique contenant la plus longue tâche, puis la clique présentant la plus petite différence entre le nombre d'employés disponibles et le nombre de tâches à affecter, puis la première clique.

TABLE 5.3 – $\mathcal{S}c$: stratégies d'exploration des cliques

Algorithme 5.1: Phase 2 : affectation des tâches, à horaires de travail fixés

Données :
FJSP, une instance du FJSP (horaires fixés)
$s_c \in \mathcal{S}c$, la stratégie de choix de clique
$s_p \in \mathcal{S}p$, la stratégie de pondération des arêtes

```
1  fonction resoudreFJSP(FJSP, s_c, s_p)
2      A ← ∅                                  //Initialiser l'affectation
3      C ← trouverCliquesMax(FJSP)            //cf. Algorithme 4.1, p. 65
4      tant que (C ≠ ∅) faire
5          K ← selectionnerClique(s_c)        //Selon la stratégie s_c
6          G(K) ← grapheAffectation(s_p, K)   //Selon la stratégie s_p et l'affectation actuelle
7          A_c ← algoHongrois(G(K))           //Obtenir une affectation de poids maximum
8          A_c ← pauseDéjeuner(A_c)           //Retirer les affectations non valides
9          A ← A ∪ A_c
10         C ← retirerTâchesAffectées(A_c)
11     retourner A
```

alors à affecter les tâches aux employés. Puisque cette dernière étape aborde le même problème que la deuxième phase de M2P, nous utiliserons la même méthode de résolution, détaillée à la Section 5.2.3. La seule différence est que nous utilisons successivement chacune des 45 variantes de manière à trouver une bonne solution initiale, alors que seule la variante la plus adaptée est utilisée en cours de résolution (*cf.* Algorithme 5.2). Par conséquent, l'utilisation successive des 45 variantes permet également de sélectionner la stratégie la plus adaptée.

Algorithme 5.2: Construction de l'affectation initiale

Données :
FJSP, une instance du FJSP
$\mathcal{S}c$, stratégies de choix de clique
$\mathcal{S}p$, stratégies de pondération des arêtes

```
1 fonction construireAffectationInitiale(FJSP, Sc, Sp)
2     A_G ← ∅                                   //Initialiser l'affectation
3     pour (s_c ∈ Sc, s_p ∈ Sp) faire
4         A_L ← resoudreFJSP(FJSP, s_c, s_p)    //Créer un couplage avec s_c et s_p
5         si (A_L meilleure que A_G) alors
6             A_G ← A_L
7             s_c* ← s_c
8             s_p* ← s_p
9     retourner (A_G, s_c*, s_p*)
```

Nous nous focalisons à présent sur le problème de construction des horaires initiaux des employés. Pour résoudre ce problème, nous proposons une heuristique de liste (*cf.* Algorithme 5.3). Cette heuristique, illustrée à la Figure 5.3, se décompose en trois étapes principales :

Ligne 3 : Une première étape consiste tout d'abord à trier les employés et les tâches de manière pertinente : les employés sont triés par criticité croissante, alors que les tâches sont triées par durées décroissantes (*cf.* Figure 5.3a). Nous utilisons ici la version « globale » de la criticité utilisée par l'opérateur *DCritiQ*. Plus précisément, nous considérons qu'un employé est d'autant plus critique que la somme des durées des tâches qui lui sont affectables est faible au regard de ses disponibilités horaire. Autrement dit, un employé est d'autant plus critique qu'il maîtrise peu de compétence et que les compétences maîtrisées ne lui donnent accès qu'à une faible charge de travail potentiel. De cette manière, les premiers horaires de travail créés portent sur les tâches les plus longues et sur les employés dont la charge de travail potentielle est la plus faible au regard de leurs disponibilités.

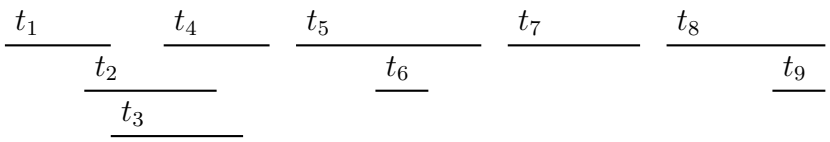
Lignes 5 à 9 : La deuxième étape consiste à associer au premier employé possible un horaire de travail permettant de contenir la première tâche non traitée. Quand une tâche est associée à un horaire, toutes les tâches également incluses dans cet horaire et susceptibles d'être assignées au même employé sont retirées de la liste des tâches à traiter. De cette manière, nous avons créé un horaire de travail susceptible de contenir plusieurs tâches pour un même employé (*cf.* Figures 5.3b à 5.3d).

Lignes 10 à 20 : Il est aisé de voir que les horaires ainsi créés ne permettent pas forcément de couvrir toutes les tâches. Par exemple, deux tâches se chevauchant complètement et requérant la même compétence ne conduisent qu'à un seul horaire de travail, alors qu'il faudra les affecter à des employés différents. La troisième étape de l'algorithme aborde ce problème en essayant de créer de nouveaux horaires de travail de manière à couvrir le profil de la demande, donné par la fonction P_d qui à tout instant t associe le nombre de tâches $P_d(t)$ s'exécutant en parallèle à cet instant-là (*cf.* Figure 5.3e).

Algorithme 5.3: Construction des horaires initiaux du personnel

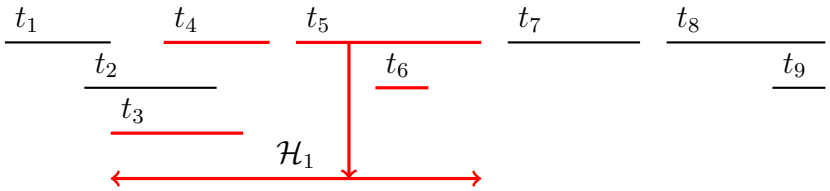
Données :
SDPTSP, une instance du SDPTSP

```
1  fonction construireHorairesInitiaux(SDPTSP)
2      H ← ∅                                   //Initialisation des horaires du personnel
3      (N, T) ← trierDonnées(SDPTSP)
4      tant que (T ≠ ∅) faire
5          t ← retirerPremierElement(T)
6          n ← premierEmployeAssignable(t, N)
7          h ← creerHoraire(t, n)
8          H ← H ∪ h
9          T ← retirerTachesAssignables(T, h, n)
10     P_d ← creerProfilDemande(SDPTSP)        //Profil du nombre de tâches
11     P_r ← creerProfilRessource(H)           //Profil du nombre d'horaires
12     C ← {x|P_d(x) > P_r(x)}
13     tant que (C ≠ ∅) faire
14         c ← retirerPremierElement(C)
15         n ← premierEmployeAssignable(c, N)
16         si (n ≠ null) alors
17             h ← creerHoraire(c, n)
18             H ← H ∪ h
19         mettreAJour(C)
20     retourner H
```



$\mathcal{T}_{n_1} : \{t_5, t_3, t_7, t_4, t_6\}$
$\mathcal{T}_{n_3} : \{t_8, t_7, t_1, t_4, t_6, t_9\}$
$\mathcal{T}_{n_2} : \{t_5, t_8, t_2, t_3, t_4, t_6, t_9\}$
$\mathcal{T}_{n_4} : \{t_5, t_8, t_2, t_3, t_7, t_1, t_4, t_6, t_9\}$

(a) Étape 1 : tri des tâches et des employés.

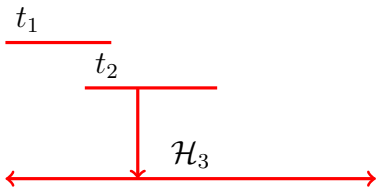


$\mathcal{H}_1 : \{t_5, t_3, t_4, t_6\}$
$\mathcal{T}_{n_3} : \{t_8, t_7, t_1, \cancel{t_4}, \cancel{t_6}, t_9\}$
$\mathcal{T}_{n_2} : \{\cancel{t_5}, t_8, t_2, \cancel{t_3}, t_4, \cancel{t_6}, t_1, t_9\}$
$\mathcal{T}_{n_4} : \{\cancel{t_5}, t_8, t_2, \cancel{t_3}, t_7, t_1, t_4, \cancel{t_6}, t_9\}$

(b) Étape 2 (itération 1) : création d'horaires pour les plus grandes tâches des employés les plus critiques.

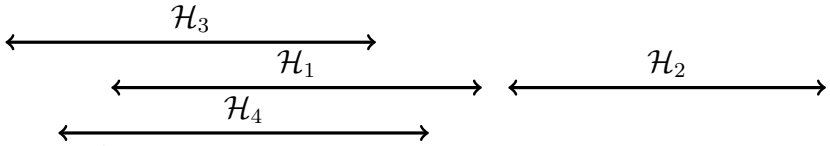
$\mathcal{H}_1 : \{t_5, t_3, t_4, t_6\}$
$\mathcal{H}_2 : \{t_8, t_7, t_9\}$
$\mathcal{T}_{n_2} : \{\cancel{t_8}, t_2, t_1, \cancel{t_9}\}$
$\mathcal{T}_{n_4} : \{\cancel{t_8}, t_2, \cancel{t_7}, t_1, \cancel{t_9}\}$

(c) Étape 2 (itération 2) : création d'horaires pour les plus grandes tâches des employés les plus critiques.



$\mathcal{H}_1 : \{t_5, t_3, t_4, t_6\}$
$\mathcal{H}_2 : \{t_8, t_7, t_9\}$
$\mathcal{H}_3 : \{t_2, t_1\}$
$\mathcal{T}_{n_4} : \{\cancel{t_2}, \cancel{t_1}\}$

(d) Étape 2 (itération 3) : création d'horaires pour les plus grandes tâches des employés les plus critiques.



(e) Étape 3 : ajout de nouveaux horaires, selon le profil $P_d(t)$.

FIGURE 5.3 – Exemple d'utilisation de l'Algorithme 5.2 sur un horizon d'un jour.

5.2.5 Fonctionnement détaillé de M2P

Nous avons vu aux sections précédentes comment obtenir une solution initiale (*cf.* Section 5.2.4), comment obtenir un voisinage sur les horaires du personnel (*cf.* Section 5.2.2) et comment obtenir une affectation des tâches aux employés (*cf.* Section 5.2.3). Nous avons également brièvement décrit l'imbrication de ces différentes étapes à la Section 5.2.1. Nous détaillons à présent le fonctionnement complet de M2P, donné à l'Algorithme 5.4 :

Lignes 2 à 5 : Les Algorithmes 5.3 et 5.1 permettent d'initialiser la méthode. Comme nous l'avons vu à la Section 5.2.4, parmi les 45 variantes de cet algorithme (9 stratégies de pondération des arêtes et 5 stratégies d'exploration des cliques), celle conduisant à la meilleure solution initiale est conservée pour la suite de la résolution.

Lignes 7 à 9 : Les différents opérateurs de voisinage présentés à la Section 5.2.2 sont alors utilisés à partir de la meilleure solution locale afin d'obtenir de nouveaux horaires prometteurs. Chacun de ces horaires donne lieu à la résolution d'un FJSP (*cf.* Algorithme 5.1) selon les stratégies de choix de clique et de pondérations des arêtes retenues à l'initialisation de la méthode.

Ligne 11 : Remarquons qu'une fois que toutes les cliques ont été visitées, il peut rester des tâches non affectées, car les couplages trouvés ne couvrent pas forcément toutes les tâches. Cela peut notamment être dû à un problème au niveau des horaires de travail du personnel. Par conséquent, suite à l'Algorithme 5.1, une descente locale tente de réduire le nombre de tâches non affectées en rallongeant les horaires de travail des employés, ou en leur ajoutant un nouvel horaire de travail. Les horaires de travail d'un employé ne sont modifiés que lorsque cela permet de couvrir complètement l'une des tâches non affectées, sans violer aucune contrainte légale ou organisationnelle. Les horaires de travail déjà existants sont rallongés au minimum, alors que les horaires ajoutés sont générés aléatoirement entre un horaire maximum, déduit des contraintes, et un horaire minimum, déduit de la tâche qui est à l'origine de ce nouvel horaire.

Ligne 19 : Notons qu'au fil de la résolution, les horaires des employés s'allongent, notamment en raison de l'opérateur *Étendre*(ρ), ce qui conduit à une perte de flexibilité en raison des contraintes légales. Par conséquent, nous proposons de réduire les horaires de travail des employés à leur minimum toutes les γ itérations. Les horaires de travail minimum des employés correspondent aux horaires minimum permettant de couvrir les tâches qui leur sont assignées. Cette réduction consiste donc simplement à recalculer les horaires des employés directement à partir de la meilleure affectation locale tâche-employé.

Ligne 25 : Afin d'éviter les optima locaux, nous proposons également une procédure de diversification consistant à opter pour une autre variante de la méthode d'affectation des tâches, en changeant les stratégies $\mathcal{S}c$ et $\mathcal{S}p$ dans l'Algorithme 5.1 au bout de δ itérations consécutives non améliorantes.

Ligne 27 : En fin de résolution, la procédure H_Δ permet de réduire l'inéquité de la meilleure solution obtenue.

5.3 Paramétrage de M2P

Nous présentons dans cette section les résultats de M2P[1]. Nous détaillons et justifions tout d'abord le paramétrage des deux phases. Par la suite, nous évaluons la qualité globale des résultats obtenus et plus particulièrement la qualité des solutions initiales, l'impact de H_Δ sur l'inéquité des solutions et sur l'efficacité des deux phases.

[1] Les algorithmes sont implémentés en JAVA et les tests sont réalisés sur un Intel Core i3-540 (3.06 GHz & 8G RAM)

Algorithme 5.4: Structure détaillée de M2P

Données :
SDPTSP||U,Δ, une instance du SDPTSP||U,Δ
$\mathcal{S}c$, les stratégies de choix de clique
$\mathcal{S}p$, les stratégies de pondération des arêtes
γ, paramètre contrôlant la réduction des horaires du personnel
δ, paramètre contrôlant le changement de $s_c{}^*$ et $s_p{}^*$

```
1  fonction M2P(SDPTSP||U,Δ, Sc, Sp)
2      H_I ← horairesInitiaux(SDPTSP||U,Δ)                    //cf. Algorithme 5.3
3      (A_I, s_c*, s_p*) ← affectationInitiale(H_I, Sc, Sp)   //cf. Algorithme 5.2
4      (A_G, A_L, H_G, H_L) ← (A_I, A_I, H_I, H_I)
5      (nbStag, nbIter) ← (0, 0)
6      tant que (temps de calcul non écoulé) faire
7          H_V ← genererVoisinage(H_L)                        //cf. Table 5.1
8          stagnation ← vrai
9          pour (H ∈ H_V) faire
10             A ← resoudreFJSP(H, s_c*, s_p*)                //cf. Algorithme 5.1
11             A ← descenteLocale(A)                          //Compléter le couplage
12             si (A meilleure que A_L) alors
13                 (A_L, H_L) ← (A, H)
14             si (A meilleure que A_G) alors
15                 (A_G, H_G) ← (A, H)
16                 stagnation ← faux
17         nbIter ← nbIter + 1
18         si (nbIter%γ = 0) alors
19             reduireHoraires(A_L, H_L)
20         si (stagnation) alors
21             nbStag ← nbStag + 1
22         sinon
23             nbStag ← 0
24         si (nbStag = δ) alors
25             (s_c*, s_p*) ← changerDeStrategie(Sc, Sp, s_c*, s_p*)   //Modifier l'Algorithme 5.1
26             nbStag ← 0
27     A_G ← H_Δ(SDPTSP||U,Δ, A_G)
28     retourner A_G
```

5.3.1 Paramétrage de la phase 1

Nous nous intéressons ici au paramétrage des opérateurs de voisinage présentés à la Section 5.2.2, p. 79. Nous avons tout d'abord réalisé quelques tests préliminaires afin de vérifier la viabilité des différents opérateurs. Cette première phase de test nous a permis d'éliminer l'opérateur de voisinage *Déplacer* qui était dominé par les autres opérateurs. Par ailleurs, nous avons également pu déterminer la taille et la composition les plus efficaces des voisinages. Rappelons en effet que chacun des opérateurs de voisinage permet d'obtenir un seul voisin. Pour obtenir un voisinage, l'opérateur o est donc utilisé μ_o fois (un même opérateur utilisé plusieurs fois conduit à des voisins différents). Les résultats obtenus montrent qu'il est plus efficace de construire davantage de voisins avec les opérateurs génériques *Ajouter*(ρ), *Étendre*(ρ) et *Retirer*(ρ) qu'avec les opérateurs dédiés *Équilibrer* et *Couvrir*. Au final, nous avons fixé $\mu_{Ajouter(\rho)}$, $\mu_{Étendre(\rho)}$ et $\mu_{Retirer(\rho)}$ à 10 contre 5 pour $\mu_{Équilibrer}$ et $\mu_{Couvrir}$. Cette configuration conduit à explorer 40 voisins à chaque itération. Par la suite, nous avons fait varier les paramètres ρ et γ de manière croisée.

- Rappelons que plus le paramètre ρ est proche de 0, plus les opérateurs *Ajouter*(ρ), *Étendre*(ρ) et *Retirer*(ρ) visent à réduire l'inéquité entre les employés. A contrario, lorsque ce paramètre est proche de 1, ces opérateurs permettent de diversifier les horaires du personnel. Ce paramètre a été testé pour les valeurs 0.2, 0.6 et 1.

- Rappelons que plus le paramètre γ est proche de 1, plus la fréquence de réduction des horaires de travail des employés est élevée, conduisant à redonner de la flexibilité lors de la recherche de nouveaux voisinages. Ce paramètre a été testé pour les valeurs 5, 10 et 50.

		γ			Moyenne
		5	10	50	
ρ	0.2	536	538	535	536,33
	0.6	538	540	534	537,33
	1	538	**544**	537	539,66
Moyenne		537,33	540,66	535,33	537,77

TABLE 5.4 – Nombre de solutions complètes selon les valeurs de ρ et γ (5 min)

		γ			Moyenne
		5	10	50	
ρ	0.2	37,6	38,5	**37,5**	37,9
	0.6	40,0	39,9	39,5	39,8
	1	40,2	40,3	41,1	40,5
Moyenne		39,3	39,6	39,4	39,4

TABLE 5.5 – Équité moyenne des solutions complètes selon les valeurs de ρ et γ (5 min)

Les Tables 5.4 et 5.5 donnent respectivement le nombre moyen de solutions complètes (sur un total de 618, 102 instances étant infaisables *cf.* Section 4.4.1, p. 75) ainsi que l'équité moyenne correspondante (en minutes), lorsque les paramètres ρ et γ varient. En ce qui concerne le paramètre ρ, il semble plus efficace, du point de vue de la réalisabilité, de prendre une valeur élevée, avec un maximum d'efficacité pour $\rho = 1$. Cela souligne l'importance de diversifier les horaires de travail des employés. En revanche, l'inéquité est d'autant plus grande que ρ est grand. Bien que cette tendance soit assez claire, le différentiel d'équité reste en moyenne assez faible. Le paramètre γ, quant à lui, n'a presque pas d'effet sur l'inéquité des solutions. En revanche, la configuration $\gamma = 10$ permet d'obtenir davantage de solutions complètes. Au final, la meilleure configuration ($\rho = 1$ et $\gamma = 10$) permet d'obtenir 544 solutions complètes présentant une inéquité moyenne d'environ 40 minutes, contre 534 solutions complètes dans la pire des configurations. Autrement dit, le nombre de solutions complètes varie relativement peu entre les différentes configurations, ce qui signifie qu'il est sans doute inutile de chercher un paramétrage plus efficace de ρ et γ.

5.3.2 Paramétrage de la phase 2

Nous nous intéressons ici aux stratégies d'exploration des cliques ($\mathcal{S}c$) et de pondération des arêtes ($\mathcal{S}p$), présentées à la Section 5.2.3, p. 80. Des tests préliminaires nous ont tout d'abord conduits à éliminer les stratégies *Densité* et *Durée*, ainsi que les stratégies *NCritiQ*, *DCritiQ*, *3.Charge+Pot*, *10.Charge+Pot* et *20.Charge+Pot*, en raison de la faiblesse de leurs résultats. Nous avons alors testé l'impact du paramètre λ qui donne le nombre d'itérations consécutives non améliorantes au bout duquel les stratégies de choix de clique et de pondération des arêtes sont modifiées. Ce paramètre a été testé pour les valeurs 1, 5, 10 et 50.

En ce qui concerne le nombre de solutions complètes, les meilleurs résultats sont obtenus pour $\lambda = 5$, avec 544 solutions complètes, contre respectivement 534 et 541 avec $\lambda = 50$ et $\lambda = 1$. Autrement dit, il vaut mieux modifier fréquemment $\mathcal{S}p$ et $\mathcal{S}c$ pour trouver davantage de solutions complètes. En ce qui concerne l'inéquité des solutions complètes, les paramétrage $\lambda = 5$ et $\lambda = 10$ conduisent tous deux à une inéquité moyenne d'environ 40 minutes contre 50 avec $\lambda = 50$. Dans l'ensemble, il est donc plus efficace de modifier fréquemment $\mathcal{S}p$ et $\mathcal{S}c$, que ce soit pour trouver plus de solutions complètes ou pour trouver des solutions plus équitables.

I_v	λ	I_t				
		100	200	300	400	Total
600	1	53	59	58	59	229
	5	53	59	58	59	229
	10	53	59	58	59	229
	50	53	59	58	59	229
800	1	41	50	53	55	199
	5	42	50	53	55	200
	10	42	50	53	55	200
	50	41	50	53	55	199
1 000	1	11	21	41	40	113
	5	11	22	42	40	115
	10	9	22	39	40	110
	50	9	19	38	40	106
Total	1	105	130	152	154	541
	5	106	131	153	154	544
	10	104	131	150	154	539
	50	103	128	149	154	534

TABLE 5.6 – Nombre de solutions complètes selon λ

I_v	λ	I_t				
		100	200	300	400	Total
600	1	30	38	41	53	41
	5	28	34	40	47	38
	10	27	33	41	60	40
	50	24	31	46	88	48
800	1	32	38	42	50	41
	5	35	35	38	44	38
	10	29	33	39	47	37
	50	27	33	39	66	42
1 000	1	43	48	50	59	52
	5	72	42	46	51	49
	10	29	54	46	49	47
	50	34	44	56	69	57
Total	1	32	39	44	53	43
	5	35	36	41	47	40
	10	28	36	41	52	41
	50	26	34	46	75	48

TABLE 5.7 – Équité moyenne des solutions complètes selon λ

5.4 Analyse des résultats expérimentaux

Nous présentons dans cette section les résultats de la méthode en deux phases. Pour évaluer l'intérêt de M2P, nous donnons tout d'abord les résultats obtenus avec la meilleure configuration ($\rho = 1$, $\gamma = 10$ et $\lambda = 5$). Par la suite, nous nous efforçons de mettre en lumière le rôle joué par chacun des composants de M2P (solution initiale, phase 1 et phase 2). Pour finir, nous mesurons l'impact de la descente locale H_Δ sur l'inéquité des solutions obtenues avec M2P. Pour évaluer la qualité des résultats, nous prendrons en compte plusieurs indicateurs :

C : le nombre total d'instances pour lesquelles la méthode trouve une solution complète.
I : l'inéquité moyenne des solutions complètes (en minutes).

A : le pourcentage moyen de tâches affectées pour les solutions partielles uniquement.
T : le temps moyen d'obtention des meilleures solutions (en secondes).

5.4.1 Résultats globaux

La meilleure configuration permet d'obtenir de très bons résultats (*cf.* Table 5.8). De manière générale, M2P est en effet capable d'obtenir 544 solutions complètes sur un maximum de 618, soit 88% de succès vis-à-vis de la réalisabilité. Plus précisément, M2P trouve une solution complète pour 67% des instances les plus chargées ($I_v = 1\,000$), contre 98% pour les instances les moins chargées ($I_v = 600$). Lorsque la méthode n'est pas capable de trouver une solution complète, le nombre de tâches affectées tourne en moyenne autour de 98%, ce qui permettrait à un décideur d'évaluer finement les besoins en intérimaires. Par ailleurs, l'équité moyenne est également très satisfaisante puisqu'elle atteint 40 minutes, ce qui est tout à fait acceptable du point de vue industriel. Sans surprise, l'équité obtenue est d'autant meilleure que l'instance est peu chargée et de taille modeste ($I_t = 100$ et $I_v = 600$), ce qui correspond aux instances les plus faciles. A contrario, les instances très chargées et de taille modeste ($I_t = 100$ et $I_v = 1\,000$) sont beaucoup plus difficiles à résoudre, à la fois du point de la réalisabilité des instances et du point de vue de l'équité entre les employés. Comparé au paramètre I_v, l'impact du paramètre I_c sur la réalisabilité des instances est beaucoup moins net. Plus précisément, M2P trouve 87,99% des solutions complètes pour les instances sans compétences rares ($I_c = I_{cc}$), contre 88,06% pour les instances avec compétences rares ($I_c = I_{cr}$).

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	42/47	11/21	106/122
	I	28	35	72	35
	A	97,50	97,78	96,71	97,05
	T	145	155	167	156
200	C	59/60	50/55	22/35	131/150
	I	34	35	42	36
	A	99,00	98,60	97,72	97,93
	T	166	138	156	154
300	C	58/58	53/58	42/56	153/172
	I	40	38	46	41
	A	99,67	99,19	98,76	98,94
	T	191	186	174	184
400	C	59/59	55/59	40/56	154/174
	I	47	44	51	47
	A	99,75	99,70	99,01	99,17
	T	236	202	196	211
Total	C	229/231	200/219	115/168	**544/618**
	I	38	38	49	**40**
	A	98,28	98,47	97,68	**97,90**
	T	184	170	173	**176**

TABLE 5.8 – Solutions finales de M2P (5 min)

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	37/54	3/47	0/21	40/122
	I	1008	755	-	989
	A	97,78	96,54	91,30	94,50
	T	0,31	0,32	0,32	0,32
200	C	55/60	20/55	2/35	77/150
	I	1229	1004	934	1163
	A	99,00	98,40	94,70	96,34
	T	0,79	0,78	0,8	0,79
300	C	56/58	30/58	0/56	86/172
	I	1382	1114	-	1289
	A	99,42	98,88	97,42	97,97
	T	1,06	1,1	1,14	1,1
400	C	58/59	42/59	3/56	103/174
	I	1456	1207	989	1341
	A	99,75	99,18	98,11	98,41
	T	1,44	1,44	1,43	1,44
Total	C	206/231	95/219	5/168	**306/618**
	I	1295	1121	967	**1235**
	A	98,27	97,87	95,35	**96,94**
	T	0,9	0,91	0,92	**0,91**

TABLE 5.9 – Solutions initiales de M2P

5.4.2 Qualité de la solution initiale

Afin d'évaluer l'intérêt général du fonctionnement itératif de M2P, nous évaluons à présent la qualité des résultats obtenus suite à la phase d'initialisation de M2P(*cf.* Table 5.9). Remarquons que certains jeux de données sont déjà presque entièrement résolus du point de vue de la réalisabilité. Plus

I_t	I_v			
	600	800	1 000	Total
100	31-28	39-35	81-72	39-35
200	39-34	43-35	59-42	44-36
300	53-40	51-38	69-46	57-41
400	75-47	74-44	91-51	79-47
Total	50-38	53-38	76-49	57-40

TABLE 5.10 – Équité moyenne avant H_Δ - après H_Δ (en minutes)

précisément, dès la solution initiale, 89% des instances les moins chargées ($I_v = 600$) présentent déjà une solution complète. En revanche, seules 5 solutions complètes sont obtenues pour les instances les plus chargées ($I_v = 1\ 000$), ce qui montre clairement l'impact du paramètre I_v. Par ailleurs, le pourcentage moyen de tâches assignées tourne autour de 96% à l'initialisation, ce qui explique en partie le gain réalisé sur le nombre de solutions complètes entre les solutions initiales et finales. De plus, l'équité moyenne est de 1235 minutes à l'initialisation contre environ 60 minutes avant H_Δ pour les solutions finales, soit un facteur 20 en moyenne. Cela montre clairement l'efficacité de l'approche en termes de réalisabilité et d'équité. Étant donnée la rapidité de cette phase d'initialisation, la qualité de la solution initiale nous semble très satisfaisante, surtout en ce qui concerne la réalisabilité de l'instance.

5.4.3 Impact de H_Δ

Nous avons vu à la Section 5.4.1 que M2P permettait d'obtenir de très bons résultats en terme d'équité. Ces résultats correspondent à l'utilisation successive de la méthode M2P à proprement parler, ainsi que de la descente locale H_Δ. Il semble par conséquent légitime d'analyser l'impact de H_Δ afin d'identifier clairement la qualité des résultats obtenus uniquement par M2P. La Table 5.10 donne la valeur moyenne de l'inéquité des solutions complètes (en minutes) avant et après la descente locale H_Δ. Il est clair que cette procédure est efficace, notamment sur les instances de grande taille. Le gain moyen ainsi obtenu est de 4 minutes sur les petites instances (soit 10% en moyenne), contre 32 minutes sur les plus grandes instances (soit 40% en moyenne). L'équité obtenue directement via M2P reste cependant de bonne qualité.

5.4.4 Impact des différents opérateurs de voisinage (Phase 1)

Nous nous intéressons à présent aux opérateurs de voisinage et cherchons à évaluer leur capacité à trouver de bonnes solutions. Pour cela, nous comparons le pourcentage moyen de meilleures solutions locales obtenues par chaque opérateur, rapporté au nombre de voisins explorés (*cf.* lignes 12 et 13 de l'Algorithme 5.4). Rappelons en effet que les opérateurs *Ajouter*(ρ), *Étendre*(ρ) et *Retirer*(ρ) explorent chacun 10 voisins à chaque itération, contre 5 pour les opérateurs *Couvrir* et *Équilibrer*. Les résultats sont récapitulés à la Table 5.11. Il est clair que les opérateurs *Ajouter*(ρ) et *Étendre*(ρ) contribuent davantage à améliorer la solution courante que les autres opérateurs. Le pourcentage moyen de succès des opérateurs *Ajouter*(ρ) et *Étendre*(ρ) est respectivement de 23% et 28%, contre 13, 17 et 19% pour les opérateurs *Retirer*(ρ), *Couvrir* et *Équilibrer*. Remarquons que l'opérateur *Retirer*(ρ) semble légèrement plus pertinent pour les instances les plus chargées ($I_v = 1\ 000$), ce qui traduit le fait qu'une instance chargée requiert une remise en question plus importante des horaires courants, soulignant ainsi la difficulté de positionner correctement les horaires du personnel dans ce cas.

I_v	Opérateur	I_t				
		100	200	300	400	Total
600	*Ajouter*(ρ)	16,6	15,5	24,4	**45,6**	**25,5**
	Étendre(ρ)	**27,5**	**31,5**	**27,9**	13,9	25,2
	Retirer(ρ)	13,8	12,6	10,0	6,8	10,8
	Couvrir	20,3	20,2	17,2	12,4	17,5
	Équilibrer	21,9	20,2	20,5	21,3	21,0
800	*Ajouter*(ρ)	20,2	15,7	17,8	26,1	20,0
	Étendre(ρ)	**27,1**	**30,2**	**31,9**	**26,8**	**29,0**
	Retirer(ρ)	17,3	14,7	13,0	11,4	14,1
	Couvrir	15,9	19,1	18,1	15,8	17,2
	Équilibrer	19,5	20,3	19,3	19,8	19,7
1 000	*Ajouter*(ρ)	**32,8**	23,8	16,0	17,9	22,6
	Étendre(ρ)	26,8	**25,8**	**32,3**	**31,2**	**29,0**
	Retirer(ρ)	15,8	15,7	15,4	14,0	15,2
	Couvrir	10,4	17,7	18,6	19,7	16,6
	Équilibrer	14,3	17,0	17,6	17,3	16,5
Total	*Ajouter*(ρ)	23,2	18,3	19,4	**29,9**	22,7
	Étendre(ρ)	**27,1**	**29,2**	**30,7**	24,0	**27,7**
	Retirer(ρ)	15,6	14,3	12,8	10,7	13,4
	Couvrir	15,5	19,0	18,0	16,0	17,1
	Équilibrer	18,5	19,2	19,1	19,4	19,1

TABLE 5.11 – Pourcentage moyen de meilleures solutions locales obtenues par opérateur de voisinage

5.4.5 Efficacité des différentes stratégies $\mathcal{S}c$ et $\mathcal{S}p$ (Phase 2)

La Table 5.12 donne le pourcentage moyen de sélection des variantes de l'Algorithme 5.1, qui interviennent pour obtenir la solution initiale ainsi que toutes les δ itérations non améliorantes (*cf.* lignes 3 et 25 de l'Algorithme 5.4). Remarquons tout d'abord que les variantes fondées sur les stratégies *DensitéDurée* et *DuréeDensité* sont sélectionnées deux fois plus souvent que *Chronologique*, ce qui signifie que la notion de clique « Dense », sur laquelle se fondent ces stratégies, est tout à fait pertinente. Notons également que les variantes fondées sur la stratégie *Charge+NCritiQ* correspondent à plus de 40% des sélections. Globalement, les combinaisons *DensitéDurée-Charge+NCritiQ* et *DuréeDensité-Charge+NCritiQ* sont bien plus efficaces que les autres variantes, ce qui souligne l'intérêt des notions de cliques « denses » et d'employés « critiques ». Bien que l'efficacité de ces différentes stratégies soit très variables, rappelons que la configuration $\delta = 5$ donne de meilleurs résultats que les configurations $\delta = 10$ ou $\delta = 50$, ce qui signifie qu'il est efficace de changer souvent de stratégie. Autrement dit, même si certaines stratégies semblent peu efficaces, il est probable qu'elles contribuent néanmoins à diversifier les solutions, conduisant au final à de meilleurs résultats.

Exploration des cliques ($\mathcal{S}c$)	Pondération des arêtes ($\mathcal{S}p$)			
	Charge	*Charge+Pot*	*Charge+NCritiQ*	*Charge+DCritiQ*
Chronologique	5,2	3,2	3.8	5.0
DuréeDensité	11,1	2,2	**20,3**	9,0
DensitéDurée	11,1	1,1	19,1	8,7

TABLE 5.12 – Pourcentage moyen de sélection des variantes de l'Algorithme 5.1

5.5 Conclusion

Nous avons présenté dans ce chapitre une méthode en deux phases exploitant la structure du problème afin de simplifier le problème d'affectation des tâches. Cette décomposition, classique dans la littérature, nous permet de gérer séparément les contraintes légales et les contraintes organisationnelles. En effet, les premières portent principalement sur les horaires de travail, et sont donc traitées en phase 1, alors que les secondes portent sur l'affectation des tâches et sont traitées durant la phase 2. Nous avons proposé un ensemble d'opérateurs de voisinage permettant de faire évoluer les horaires du personnel selon l'affectation courante des tâches de manière à mieux couvrir les tâches non affectées, ou à équilibrer la charge de travail des employés. Les résultats obtenus sont très satisfaisants, à la fois au niveau de la réalisabilité et au niveau de l'équité. La procédure H_Δ permet d'améliorer significativement la qualité de la solution en terme d'équité, sans pour autant qu'il soit absolument nécessaire d'y recourir pour obtenir une solution convenable du point de vue industriel. Par ailleurs, les résultats expérimentaux soulignent l'importance de diversifier à la fois l'espace exploré ($\rho = 1$) et les solutions construites ($\delta = 5$). Les travaux présentés dans ce chapitre ont été soumis à International Journal of Production Research [168].

6

Une approche par contraintes

Nous présentons dans ce chapitre une modélisation du SDPTSP|U = 0|Δ sous la forme d'un Programme par Propagation de Contraintes, noté $PPC_{|U=0|\Delta}$. Contrairement à la méthode en deux phases (*cf.* Chapitre 5 p. 77), ce modèle aborde simultanément l'affectation des tâches et la définition des horaires. Le cœur du modèle $PPC_{|U=0|\Delta}$ repose sur l'utilisation de variables ensemblistes qui permettent d'exprimer de nombreuses contraintes de manière naturelle. À partir de $PPC_{|U=0|\Delta}$, nous introduisons un second modèle, noté $PPC_{||U}$ permettant de résoudre le SDPTSP||U, puis nous agrégeons les objectifs U et Δ de manière lexicographique, ce qui conduit à un troisième modèle, noté $PPC_{||U,\Delta}$. Nous présentons plusieurs stratégies de branchement et d'exploration. Une étude expérimentale permet d'évaluer l'intérêt de la méthode ainsi que l'impact des différentes stratégies de branchement et d'exploration.

Sommaire

6.1 Utilisation de la Programmation par Contraintes

La Programmation par Contraintes (PPC) constitue à la fois un langage de modélisation et un outil de résolution permettant d'aborder de nombreux problèmes combinatoires et continus. La structure générale d'un modèle PPC repose sur la définition de variables, associées à un domaine de définition et reliées les unes aux autres via un réseau de contraintes [176]. L'étude de l'état de l'art montre que la PPC permet de modéliser efficacement des problèmes complexes tout en restant très synthétique et modulable, ce qui est souhaitable dans le cadre d'un projet industriel [192]. De nombreuses études portant sur l'utilisation de la PPC montrent l'efficacité de cette approche, notamment vis-à-vis de problèmes de planification difficiles tels que la planification du personnel médical en milieu hospitalier [1, 60, 139, 150, 151, 194] ou la planification d'équipages [51, 57, 94, 109, 152]. La PPC est généralement plus efficace lorsqu'il s'agit de trouver une solution réalisable que lorsqu'il s'agit de trouver une solution optimale. Pour cette raison, de nombreux travaux proposent des méthodes hybrides alliant la PPC à d'autres méthodes telles que la recherche locale ou la génération de colonnes [26, 64, 107]. Dans cette lignée, nous combinons ici une approche PPC avec la descente locale H_Δ, présentée à la Section 4.2.3, p. 66.

6.2 Un modèle PPC pour SDPTSP|U = 0|Δ

6.2.1 Choix de la modélisation

Étant donné que nous avons opté pour une modélisation de type affectation (*cf.* Section 4.1.4, p. 63) trois formulations « classiques » sont envisageables :

1. Il est tout d'abord possible d'indiquer pour chaque employé l'ensemble des tâches qui lui sont affectées. Les variables de décision sont alors représentées sous la forme d'un vecteur X de variables ensemblistes indiquant pour chaque employé l'ensemble des tâches qui lui sont affectées. Par exemple, l'affectation des tâches t_1 et t_5 à l'employé e_1 est alors notée : $X_{e_1} = \{t_1, t_5\}$.
2. Il est également envisageable d'indiquer pour chaque tâche l'employé qui lui est affecté. Les variables de décision sont alors représentées sous la forme d'un vecteur X de variables entières indiquant pour chaque tâche, l'employé qui lui est affecté. Par exemple, l'affectation des tâches t_1 et t_5 à l'employé e_1 est alors notée : $X_{t_1} = e_1$ et $X_{t_5} = e_1$.
3. Il est enfin possible de sélectionner les couples tâche-employé correspondant à une affectation. Les variables de décision sont alors représentées sous la forme d'une matrice X de variables booléennes prenant la valeur 1 lorsqu'elles représentent un couple tâche-employé sélectionné. Par exemple, l'affectation des tâches t_1 et t_5 à l'employé e_1 est alors notée : $X_{t_1, e_1} = 1$ et $X_{t_5, e_1} = 1$.

Ces trois formulations permettent de prendre en compte la notion de tâche insécable de manière implicite et permettent également de modéliser aisément les différentes contraintes d'affectation (ubiquité, unicité, *etc.*). Remarquons cependant que la plupart des contraintes légales du SDPTSP|U = 0|Δ portent sur les plannings individuels des employés et s'expriment plus facilement à partir des ensembles de tâches affectées aux employés qu'à partir des tâches ou des couples tâches-employés. Pour cette raison, nous privilégions ici la première formulation, fondée sur l'utilisation de variables ensemblistes.

6.2.2 Variables du modèle $PPC_{|U=0|\Delta}$

Les variables du modèle $PPC_{|U=0|\Delta}$ sont présentées à la Table 6.1. L'idée directrice du modèle est de représenter l'affectation des tâches aux employés via des variables ensemblistes. Pour cela, nous

Variable	Définition
$\Delta \in \mathbb{N}$	Valeur de l'inéquité
$\forall n \in \mathcal{N}, \Delta_n \in \mathbb{Z}$	Ecart à la charge médicale idéale de l'employé n
$\forall d \in \mathcal{D}, \forall n \in \mathcal{N}^+, Tasks_{d,n} \subset \mathcal{E}_{d,n}$	Tâches affectées à l'employé n le jour d
$\boldsymbol{\forall d \in \mathcal{D}, \forall n \in \mathcal{N},}$	**Pour toute vacation d, pour tout employé n**
$First_{d,n} \in \{start_t \cup \omega, t \in \mathcal{E}\}$	Début de la vacation d pour n
$Last_{d,n} \in \{end_t \cup \alpha, t \in \mathcal{E}\}$	Fin de la vacation d pour n
$Lunch_{d,n} \in \{0, \text{LUNCHP}\}$	Durée de la pause déjeuner de n durant la vacation d
$Span_{d,n} \in [\![0; \text{SPAN}_{\text{MAX}}]\!]$	Temps présentiel de n durant la vacation d
$Work_{d,n} \in [\![0; \text{WORKD}_{\text{MAX}}]\!]$	Temps travaillé par n durant la vacation d
$Load_{d,n} \in [\![0; \text{WORKD}_{\text{MAX}}]\!]$	Charge de travail de n durant la vacation d
$\boldsymbol{\forall n \in \mathcal{N},}$	**Pour tout employé n**
$\forall i \in \{1, 2\}, Break_{i,n} \in \mathbb{Z}$	Début du $i^{\text{ème}}$ repos hebdomadaire de n
$\forall d \in [\![1; 8]\!], Off_{d,n} \in \{0, 1\}$	$= 1$ ssi n est en repos le jour d

TABLE 6.1 – Variables du modèle $\text{PPC}_{|\text{U=0}|\Delta}$

associons à chaque employé $n \in \mathcal{N}$ et à chaque jour de la semaine $d \in \mathcal{D}$ une variable ensembliste $Tasks_{d,n}$ correspondant à l'ensemble des tâches affectées à l'employé n durant le jour d. Pour simplifier l'expression de certaines contraintes, nous utilisons également les variables ensemblistes $Tasks_n$ qui donnent l'ensemble des tâches affectées à l'employé n sur toute la semaine. Les variables Δ_n donnent l'écart à la charge médicale idéale de chaque employé, ce qui permet d'obtenir la valeur globale de l'inéquité Δ. Les dates de début et de fin des vacations des employés, respectivement notées $First_{d,n}$ et $Last_{d,n}$, sont déduites des variables d'affectation au moyen de contraintes de liaisons. Ces dates nous permettent de garantir les temps de repos minimum et les temps de travail maximum des employés. Les variables $First_{d,n}$ et $Last_{d,n}$ sont par exemple utilisées pour contraindre les variables $Span_{d,n}$ qui donnent l'amplitude des vacations des employés. Les variables entières $Lunch_{d,n}$ donnent la durée de la pause déjeuner de l'employé n durant la vacation d, ce qui permet de déduire le temps travaillé des employés, noté $Work_{d,n}$. Les variables $Load_{d,n}$ donnent quant à elles la charge de travail pour chaque employé et pour chaque jour, *i.e.* la somme des durée des tâches affectées aux employés durant la vacation. Cette charge intervient au niveau de la contrainte de la pause déjeuner. Afin de positionner les vacations des employés par rapport aux repos hebdomadaires des employés, nous introduisons les variables entières $Break_{i,n}$ qui donnent la date de début du $i^{\text{ème}}$ repos hebdomadaire de l'employé n. Pour finir, les variables binaires $Off_{d,n}$ permettent de comptabiliser les jours de repos des employés.

6.2.3 Contraintes du modèle $\text{PPC}_{|\text{U=0}|\Delta}$

Certaines contraintes font directement référence à l'organisation interne de l'entreprise ou à la législation en vigueur, nous parlerons alors de *contraintes organisationnelles* ou de *contraintes légales*, alors que d'autres contraintes sont utilisées pour des raisons de cohérence entre les variables, nous parlerons alors de *contraintes de liaison*. Pour finir, certaines contraintes peuvent être exprimées directement au niveau des domaines des variables. Nous parlerons alors de *contraintes de prétraitement*, afin de mettre en valeur le fait qu'elles n'engendrent aucun coût de calcul supplémentaire durant la résolution. Nous présentons dans la suite les contraintes du modèle $\text{PPC}_{|\text{U=0}|\Delta}$. Afin d'en faciliter la compréhension, les différentes contraintes nécessaires à la modélisation du problème sont présentées

sous leur forme mathématique, leur notation PPC, donnée entre parenthèses, renvoie à un lexique des contraintes utilisées (*cf.* Annexe A).

Contraintes de prétraitement

Les tâches obligatoires $\mathcal{M}_{d,n}$ de l'employé n sont tout d'abord intégrées au noyau de la variable $Tasks_{d,n}$ via les contraintes (6.3), ce qui permet de respecter la contrainte C 2. Les tâches non réalisable par l'employé n durant la vacation d ne font pas partie de l'enveloppe initiale des variables $Tasks_{d,n}$ (*cf.* Table 6.1), ce qui permet de prendre en compte les contraintes C 1, C 3 et C 7. De manière similaire, les domaines des variables de $Work_{d,n}$ et $Span_{d,n}$ sont bornés lors de leur création (*cf.* Table 6.1), ce qui permet de respecter automatiquement les contraintes C 9 et C 8. Comme nous pouvons le voir, plusieurs contraintes métier sont ainsi respectées dès la création des variables, mais de nombreuses contraintes de liaisons seront tout de même nécessaires pour que les différentes variables soient cohérentes entre elles.

Contraintes métier

Les contraintes (6.1) et (6.2) permettent de calculer la valeur de l'inéquité Δ en fonction des tâches cliniques affectées aux employés, de leur charge clinique idéale et de leur charge administrative obligatoire (*cf.* Section 4.1.3, p. 61). Les contraintes (6.4) permettent de s'assurer que les employés ne sont pas affectés à plus d'une tâche à la fois, ce qui permet de respecter la contrainte C 4. Les contraintes (6.5) permettent de s'assurer que toutes les tâches sont affectées et qu'aucune tâche n'est réalisée par plus d'un employé. Autrement dit, les tâches affectées aux employés forment une partition de l'ensemble des tâches, ce qui permet de vérifier la contrainte C 6. Les contraintes (6.6) permettent de prendre en compte la durée de la pause déjeuner lors du calcul du temps travaillé quotidiennement par chaque employé. À partir de cette donnée, il est alors aisé de borner le temps travaillé hebdomadairement par chaque employé via les contraintes (6.7), vérifiant ainsi la contrainte C 10.

La contrainte (6.8) permet de garantir le repos minimum de chaque employé entre deux journées de travail consécutives, vérifiant ainsi la contrainte C 11. Les contraintes (6.9) permettent de s'assurer qu'aucune journée de travail n'empiète sur le repos hebdomadaire. Les contraintes (6.10) permettent de positionner les deux repos hebdomadaires de manière à ce que la contrainte glissante C 12 soit respectée. Les contraintes (6.11) et (6.12) permettent de garantir le nombre minimum de jours de repos pour chaque employé, vérifiant ainsi la contrainte glissante C 13.

Les contraintes (6.13) et (6.14) permettent de garantir une pause déjeuner aux employés dont les vacations remplissent les conditions de la contrainte C 14. Plus précisément, les contraintes (6.13) garantissent que la charge totale de travail affecté à un employé ne dépasse pas son temps de présentiel travaillé, autrement dit, cela garantit que les employés ne travaillent pas durant un temps total correspondant à la durée de la pause déjeuner, alors que les contraintes (6.14) permettent de calculer la durée de la pause déjeuner en fonction des dates de début et de fin des vacations.

Contraintes de liaison

Les contraintes (6.15) permettent de déduire les tâches assignées aux employés sur la semaine à partir des tâches quotidiennes assignées aux employés. Les contraintes (6.16) et (6.17) permettent de calculer les dates de début et de fin de chaque vacation en fonction de l'ensemble des tâches qu'elles contiennent. Il n'est pas nécessaire de vérifier que cet ensemble est non vide, car les contraintes minOverSet et maxOverSet associées à notre modélisation des vacations vides, permettent de réaliser une

$\min \Delta \quad s.c.$

$$\forall (n_0, n_1) \in \mathcal{N}^2, n_0 \neq n_1 \quad \Delta \geq \Delta_{n_0} - \Delta_{n_1} \qquad \text{(geq)} \qquad (6.1)$$

$$\forall n \in \mathcal{N}, \quad \Delta_n = Load_n - target_n - \sum_{t \in \mathcal{M}_n} proc_t \qquad \text{(eq)} \qquad (6.2)$$

Contraintes organisationnelles

$$\forall n \in \mathcal{N}, \quad \mathcal{M}_{d,n} \subset Tasks_{d,n} \qquad \text{(prétraitement)} \qquad (6.3)$$

$$\forall n \in \mathcal{N}, \forall d \in \mathcal{D}, \forall \mathcal{K} \in \mathcal{C}, \quad card(\mathcal{K} \cap Tasks_{d,n}) \leq 1 \qquad \text{(card \& setInter)} \qquad (6.4)$$

$$\bigcup_{n \in \mathcal{N}, d \in \mathcal{D}} Tasks_{d,n} = \mathcal{E} \qquad \text{(partition)} \qquad (6.5)$$

Contraintes légales ($\forall n \in \mathcal{N}$)

Temps travaillé maximum quotidien/hebdomadaire

$$\forall d \in \mathcal{D}, \quad Work_{d,n} \leq Span_{d,n} - Lunch_{d,n} \qquad \text{(leq)} \qquad (6.6)$$

$$\sum_{d \in \mathcal{D}} Work_{d,n} \leq \text{WorkW}_{\text{MAX}} \qquad \text{(leq)} \qquad (6.7)$$

Repos minimum quotidien/hebdomadaire

$$\forall d \in [\![1;6]\!], \quad First_{d+1,n} - Last_{d,n} \geq \text{BreakD}_{\text{MIN}} \qquad \text{(geq)} \qquad (6.8)$$

$$\forall d \in \mathcal{D}, \forall i \in \{1,2\} \quad \begin{cases} Last_{d,n} \leq Break_{i,n} \textbf{ ou} \\ First_{d,n} \geq Break_{i,n} + \text{BreakW}_{\text{MIN}} \end{cases} \qquad \text{(or)} \qquad (6.9)$$

$$Break_{2,n} - Break_{1,n} \leq \text{Week} \qquad \text{(leq)} \qquad (6.10)$$

Jour de repos hebdomadaire

$$card(\{d \in [\![1; 7 - off_n^{-1}]\!] \,;\, Off_{d,n} = 1\}) \geq 1 \qquad \text{(count)} \qquad (6.11)$$

$$card(\{d \in [\![2;8]\!] \,;\, Off_{d,n} = 1\}) \geq 1 \qquad \text{(count)} \qquad (6.12)$$

Pause déjeuner

$$\forall d \in \mathcal{D}, \quad Load_{d,n} \leq Work_{d,n} \qquad \text{(leq)} \qquad (6.13)$$

$$\forall d \in \mathcal{D}, \quad Lunch_{d,n} = \text{LunchP} \Leftrightarrow \begin{cases} First_{d,n} \leq \text{LunchS}_d \textbf{ et} \\ Last_{d,n} \geq \text{LunchE}_d \textbf{ et} \\ Span_{d,n} \geq \text{SmallD} \end{cases} \qquad \text{(ifOnlyIf)} \qquad (6.14)$$

Contraintes de liaisons ($\forall d \in \mathcal{D}, \forall n \in \mathcal{N}$)

$$Tasks_n = \bigcup_{d \in \mathcal{D}} Tasks_{d,n} \qquad \text{(partition)} \qquad (6.15)$$

$$First_{d,n} = \min_{t \in Tasks_{d,n}} start_t \qquad \text{(minOverSet)} \qquad (6.16)$$

$$Last_{d,n} = \max_{t \in Tasks_{d,n}} end_t \qquad \text{(maxOverSet)} \qquad (6.17)$$

$$\forall d \in [\![1;8]\!], \quad Tasks_{d,n} \cap \mathcal{O}_d = \emptyset \Leftrightarrow Off_{d,n} = 1 \qquad \text{(setInter)} \qquad (6.18)$$

$$|Tasks_{d,n}| = 0 \Leftrightarrow \begin{cases} First_{d,n} = \omega \textbf{ et } Last_{d,n} = \alpha & \text{(eq)} \\ Span_{d,n} = 0 \textbf{ et } Load_{d,n} = 0 & \text{(eq)} \end{cases} \qquad \text{(card \& ifOnlyIf)} \qquad (6.19)$$

$$|Tasks_{d,n}| > 0 \Leftrightarrow \begin{cases} Span_{d,n} = Last_{d,n} - First_{d,n} & \text{(scalar)} \\ Load_{d,n} = \sum_{t \in Tasks_{d,n}} proc_t & \text{(sumOverSet)} \end{cases} \qquad \text{(card \& ifOnlyIf)} \qquad (6.20)$$

Initialisation ($\forall n \in \mathcal{N}$)

$$\text{si } work^{-1}{}_n \geq \text{DayS}_1, \text{ alors} \quad Off_{1,n} = 0 \qquad \text{(prétraitement)} \qquad (6.21)$$

$$First_{1,n} \geq \text{BreakD}_{\text{MIN}} - work_n^{-1} \qquad \text{(prétraitement)} \qquad (6.22)$$

$$Break_{1,n} \geq -work_n^{-1} \qquad \text{(prétraitement)} \qquad (6.23)$$

$$Break_{1,n} \geq -break_n^{-1} - \text{BreakW}_{\text{MIN}} \quad \textbf{et} \quad Break_{2,n} \geq \text{DayS}_2 \qquad \text{(prétraitement)} \qquad (6.24)$$

disjonction constructive (*cf.* Annexe A, p. 171). En effet, lorsqu'une tâche t est retirée de l'enveloppe de $Tasks_{d,n}$, il n'est pas nécessaire d'attendre que le noyau de $Tasks_{d,n}$ soit non vide pour conclure que les valeurs $start_t$ et end_t ne font plus partie des domaines des variables $First_{d,n}$ et $Last_{d,n}$. Si la vacation est vide, les variables $First_{d,n}$ et $Last_{d,n}$ sont alors fixées respectivement à ω et α, ce qui permet de respecter les contraintes sur l'amplitude maximale d'une vacation et le repos minimal entre deux vacations consécutives. Les contraintes (6.18) permettent d'identifier les jours de repos en fonction des tâches affectées aux employés. Les contraintes (6.19) et (6.20) permettent de gérer le cas des vacations vides. Pour finir, les contraintes (6.21) à (6.24) permettent de prendre en compte l'historique de chaque employé.

6.3 Modélisations PPC autour du SDPTSP

6.3.1 Modèles PPC pour SDPTSP||U et SDPTSP||U,Δ

À partir du modèle $\text{PPC}_{|\text{U}=0|\Delta}$, nous pouvons facilement obtenir un deuxième modèle permettant de résoudre le SDPTSP||U. Il suffit en effet de supprimer les variables Δ et Δ_n et de remplacer les contraintes (6.1) et (6.2) par les contraintes (6.25). Nous remplaçons également les contraintes (6.5) par les contraintes (6.26) qui incluent l'employé fictif. De cette manière, toutes les tâches non affectées sont réalisées par l'employé fictif, sans que cela ne viole aucune contrainte. Ces modifications nous conduisent à un nouveau modèle, noté $\text{PPC}_{||\text{U}}$. Pour résoudre le SDPTSP||U,Δ, nous considérons les objectifs Δ et U de manière lexicographique, conduisant ainsi à un troisième modèle, noté $\text{PPC}_{||\text{U},\Delta}$.

$$\text{U} = |\mathcal{E}| - |Tasks_{N+1}| \qquad \text{(card \& scalar)} \quad (6.25)$$

$$\bigcup_{n\in\mathcal{N}^+, d\in\mathcal{D}} Tasks_{d,n} = \mathcal{E} \qquad \text{(partition)} \quad (6.26)$$

6.4 Stratégies de recherche

Pour trouver de bonnes solutions à un problème donné, il est souvent important d'utiliser une stratégie de recherche adaptée au problème. En effet, la PPC est généralement plus efficace pour rechercher des solutions satisfaisant les contraintes du problème que pour en trouver l'optimum. Pour trouver une bonne solution dans un temps limité, il est fréquent de chercher tout d'abord une première solution, puis de poursuivre la recherche, en contraignant l'objectif, de manière à trouver des solutions améliorantes. Dans ce cas, nous parlerons d'approche de type *top-down*. Il est alors important de trouver une bonne solution initiale, car cela induit un filtrage plus fort lors de la recherche d'une solution améliorante.

Bien que cela ne soit pas obligatoire, il est naturel de brancher sur les variables de décision pour réduire leur domaine. Dans le cas du modèle $\text{PPC}_{|\text{U}=0|\Delta}$, cela revient à brancher sur les variables $Tasks_n$ et à choisir une valeur de l'enveloppe pour l'ajouter au noyau. Autrement dit, il s'agit de sélectionner un employé et de lui affecter une tâche qu'il est capable de réaliser. Les stratégies de choix de variable (VarOS) et de choix de valeur (ValOS) ont pour buts respectifs de choisir la variable ainsi que la valeur qui formeront le branchement. Nous proposons ci-dessous plusieurs VarOS et ValOS.

VarOS : Parmi les variables $Tasks_n$ (*i.e.* les tâches affectées aux employés), on choisit :

- **LW (Less Working)** : la variable correspondant à l'employé dont la charge médicale est la plus petite. L'idée sous-jacente est d'affecter les tâches restantes aux employés qui travaillent le moins, compte tenu de leur charge médicale idéale. Ce schéma de branchement vise à réduire l'inéquité de la solution courante.
- **MW($\bar{l}$) (Most Working($\bar{l}$))** : la variable correspondant à l'employé dont la charge médicale est la plus grande. Seuls les employés dont la charge médicale est inférieure à une certaine limite, contrôlée par un paramètre $\bar{l}$, sont considérés. Plus précisément, parmi les employés dont l'écart à la charge idéale est inférieur à $\bar{l}$, on choisit l'employé avec le plus grand écart. Si tous les employés sont au-dessus de cette limite, on choisit un employé aléatoirement. Ce schéma de branchement peut être utilisé de deux manières différentes selon la valeur de $\bar{l}$. Par exemple, lorsque $\bar{l}$ est positif cela conduit le solveur à affecter des tâches aux infirmières travaillant le plus, même lorsqu'elles ont déjà dépassé leur temps cible, ce qui permet, dans une certaine mesure, de conserver davantage de choix pour les tâches restantes et ainsi éviter les impasses. Cette stratégie ne vise pas à trouver des solutions équitables, mais à trouver davantage de solutions complètes. En revanche, lorsque $\bar{l}$ est négatif, cela conduit le solveur à affecter un temps de travail minimal à un maximum d'infirmières. Même si l'utilisation de valeurs négatives lors de la configuration du paramètre $\bar{l}$ constitue un bon moyen de répartir la charge de travail globale de manière équilibrée, cette stratégie est cependant très différente de LW qui tente d'améliorer la solution à chaque branchement, alors que MW($\bar{l}$) permet au solveur de détériorer la solution courante, en espérant qu'il soit possible de l'améliorer en fin de résolution.

ValOS Soit n_c l'indice de l'employé relatif à la variable $Tasks_{n_c}$ choisie, parmi les valeurs du domaine ouvert de $Tasks_{n_c}$ (*i.e.* dans l'enveloppe, mais pas dans le noyau) :

- **BT (Biggest Task)** : on cherche la tâche la plus longue.
- **BO (Biggest Overlap)** : on choisit la tâche appartenant au plus grand ensemble de tâches concomitantes. L'idée sous-jacente est d'affecter les tâches fixées au niveau des pics d'activité le plus rapidement possible.
- **LN (Lack of Nurse)** : on choisit la tâche présentant le plus petit nombre d'employés potentiellement affectables. L'objectif est de repérer en amont les tâches difficiles à affecter de manière à limiter le nombre de tâches non assignées.
- **BID (Best Increase of Density)** : on cherche la tâche dont l'ajout permet la plus grande augmentation de densité de travail. La densité de travail d'un jour est définie comme la charge de travail de ce jour divisée par l'amplitude de la journée de travail (la densité d'une vacation vide est initialisée à 0). L'idée sous-jacente est de compléter les journées de travail déjà commencées de manière à favoriser l'apparition de plannings compacts. Plus formellement, pour chaque tâche t potentiellement assignable à l'employé n_c, il est possible de calculer la nouvelle charge de travail $Load^{t+}_{d,n_c}$ ainsi que le temps de présence $Span^{t+}_{d,n_c}$ correspondant à l'ajout de la tâche t au jour de travail d de l'employé n_c. L'expression δ_t (6.27) permet ainsi de quantifier l'augmentation de densité produite par t et il s'agit alors de sélectionner la tâche conduisant à la plus petite valeur de δ_t.

$$\delta_t = \frac{Load_{d,n_c}}{Span_{d,n_c}} - \frac{Load^{t+}_{d,n_c}}{Span^{t+}_{d,n_c}} \in [\text{-}1; 1[\tag{6.27}$$

6.5 Résultats expérimentaux

Nous présentons dans cette section les résultats du modèle $PPC_{||U,\Delta}$[1]. Nous évaluons tout d'abord l'impact des stratégies de branchement et d'exploration à la fois sur la solution finale et sur la solution initiale. Par la suite, nous évaluons l'impacte de H_Δ sur l'inéquité et nous comparons les meilleurs résultats de $PPC_{||U,\Delta}$ par rapport à ceux obtenus avec M2P. Pour évaluer la qualité des résultats, nous prendrons en compte plusieurs indicateurs :

C : le nombre total d'instances pour lesquelles la méthode trouve une solution complète.
I : l'inéquité moyenne des solutions complètes (en minutes).
A : le pourcentage moyen de tâches affectées pour les solutions partielles uniquement.
T : le temps moyen d'obtention des meilleures solutions (en secondes).

6.5.1 Impact des stratégies de recherche

Des tests préliminaires ont tout d'abord permis de fixer le paramètre $\bar{l}$, intervenant dans la stratégie MW, à une valeur de 180 minutes, ce qui semblait efficace du point de vue du nombre de tâches non affectées. Nous évaluons à présent l'impact des stratégies de recherche sur les performances du modèle, sachant que les résultats détaillés sont donnés à l'Annexe B.1. Par ailleurs, la Figure 6.1 propose une vue d'ensemble des performances selon les différentes stratégies utilisées. Dans l'ensemble, le branchement MW permet d'obtenir plus de solutions complètes que LW, ce qui confirme l'intuition selon laquelle le fait de surcharger certains employés permet de conserver plus de latitude pour les affectations futures (*cf.* Table 6.2). Plus précisément, MW conduit en moyenne à 442 solutions complètes contre 279,4 pour LW. Ce différentiel est très marqué avec la stratégie d'exploration BID (MW :435 - LW :85), alors qu'il s'inverse légèrement avec LN (MW :477 - LW :481). Remarquons d'ailleurs que la stratégie LN conduit systématiquement aux meilleurs résultats, ce qui souligne la pertinence de cette stratégie.

	LW	MW	Moyenne
BO	348	462	405
BT	194	377	285,5
BID	85	435	260
LN	481	477	479
Moyenne	279,4	442	360,5

TABLE 6.2 – Nombre de solution complètes selon les stratégies de recherche (sur 618 instances)

	LW	MW	Moyenne
BO	134	157	147
BT	29	248	173
BID	252	172	185
LN	116	128	122
Moyenne	98	195	157

TABLE 6.3 – Inéquité des solutions complètes selon les stratégies de recherche (en minutes)

En ce qui concerne l'inéquité entre les employés, le branchement LW donne globalement de meilleurs résultats que MW, ce qui est cohérent avec l'idée générale de ces deux stratégies (*cf.* Table 6.3). Plus précisément, LW conduit en moyenne à une inéquité de 99 minutes contre 195 pour MW. Ce différentiel est très marqué avec la stratégie d'exploration BT (MW :248 - LW :29), alors qu'il s'inverse avec BID (MW :173 - LW :252).

[1] Les algorithmes sont implémentés en JAVA et nous utilisons Choco 3-1-1 [62] pour implémenter les modèles PPC. Les tests sont réalisés sur un Intel Core i3-540 (3.06 GHz & 8G RAM)

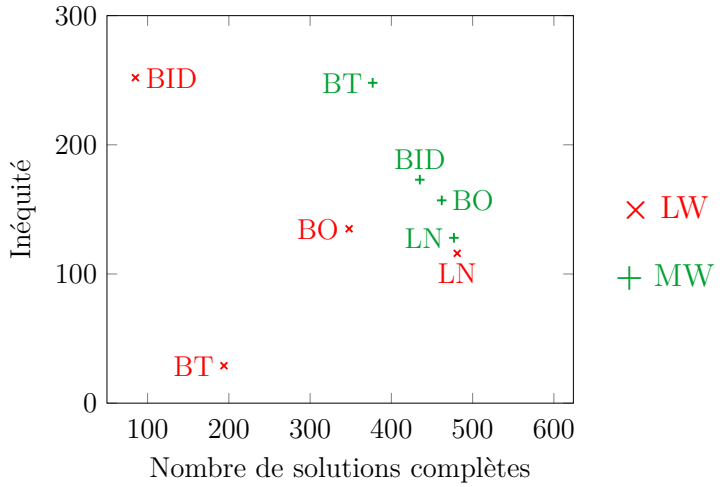


FIGURE 6.1 – Positionnement relatif des différentes stratégies.

La Table 6.4 donne le pourcentage moyen de tâches affectées pour les solutions partielles. Globalement, le pourcentage de tâches affectées est plus grand avec le branchement MW (97,77%) qu'avec le branchement LW (96,96%). Par ailleurs, le temps d'obtention de la meilleure solution est largement plus élevé avec MW qu'avec LW (*cf.* Table 6.5). Cette dernière observation nous conduit à évaluer la qualité des solutions initiales pour mieux comprendre l'origine de cette différence. Pour cela nous comparons l'évolution du nombre de solutions complètes et de leur inéquité entre la solution initiale et la solution finale. En ce qui concerne l'inéquité, nous limitons la comparaison aux instances pour lesquelles la solution initiale est complète. Les tendances observées sur les solutions initiales correspondent à celles observées sur les solutions finales. Autrement dit, MW permet d'obtenir davantage de solutions complètes (384,2 contre 269,4) mais avec une inéquité plus grande (320 contre 141). L'inéquité initiale étant assez forte avec MW, il semble cohérent que MW conduise à une réduction de l'inéquité plus importante au fil de la recherche (*cf.* Table 6.6). En appliquant le même raisonnement au nombre de solutions complètes, on pourrait s'attendre à ce que LW conduise à une forte amélioration vis-à-vis du nombre de solutions complètes, ce qui n'est pas le cas. En effet, durant la recherche, MW permet d'obtenir environ six fois plus de nouvelles solutions complètes par rapport à LW (*cf.* Table 6.7). Il est possible que l'apparition de tâches non affectées se fasse plus tardivement avec une structure de branchement de type MW qu'avec une structure de type LW. Autrement dit, la réduction du nombre de tâches non affectées avec une stratégie LW requiert de remonter plus haut dans l'arbre de recherche.

De la même manière que M2P, la méthode a plus de difficulté à trouver des solutions complètes pour les instances très chargées (I_v= 1 000) que pour les instances peu chargées (I_v= 600). Par ailleurs, la méthode semble avoir plus de difficultés à trouver des solutions améliorantes pour les instances de grandes tailles (I_t= 400) que pour les instances de petites taille (I_t= 100).

	LW	MW	Moyenne
BO	97,56	97.92	97,70
BT	96,95	97,21	97,06
BID	95,61	97,81	96,29
LN	97,56	97,83	97,70
Moyenne	96,87	97,68	97,18

TABLE 6.4 – Pourcentage de tâches affectées selon les stratégies de recherche.

	LW	MW	Moyenne
BO	75	141	108
BT	37	132	84
BID	53	124	88
LN	63	147	105
Moyenne	55	140	97

TABLE 6.5 – Temps d'obtention moyen de la meilleure solution (en secondes)

	LW	MW	Moyenne
BO	39	131	85
BT	6	150	78
BID	81	137	109
LN	21	111	66
Moyenne	30	135	83

TABLE 6.6 – Variation de l'inéquité moyenne des solutions complètes durant la résolution

	LW	MW	Moyenne
BO	9	49	29
BT	6	66	36
BID	22	87	54,5
LN	0	10	5
Moyenne	10	57,4	33,7

TABLE 6.7 – Variations du nombre de solutions complètes durant la résolution

6.5.2 Impact de H_Δ

Comme nous avons pu le voir aux Sections 4.4.2 et 5.4.3, la descente locale H_Δ est bien adaptée au SDPTSP||U,Δ et donne de bons résultats. Contrairement à la méthode en deux phases (M2P), H_Δ est ici utilisée sur chaque solution. Cela se justifie par le fait que le nombre de solutions trouvées est bien plus faible avec $PPC_{||U,\Delta}$ qu'avec M2P. Il est donc possible d'appliquer H_Δ sur chaque solution sans que cela ne porte à conséquence sur le nombre de solutions complètes que la méthode est capable d'obtenir. Pour évaluer l'impact de H_Δ sur les résultats de $PPC_{||U,\Delta}$ nous comparons à présent l'inéquité des meilleures solutions avant et après H_Δ, selon les différentes stratégies (*cf.* Table 6.8). En moyenne, H_Δ réduit l'inéquité d'un facteur 1.8 pour LW contre 5.3 avec MW. Cette descente locale permet ainsi de compenser en grande partie le différentiel d'équité entre MW et LW. Autrement dit, il est clair que cette descente locale est très efficace, notamment dans le cadre d'une méthode de résolution se focalisant sur l'obtention de solutions complètes.

	LW	MW	Moyenne
BO	253 - 134	987 - 157	672 - 147
BT	41 - 29	1090 - 248	734 - 173
BID	325 - 252	967 - 172	862 - 185
LN	235 - 116	1034 - 128	633 - 122
Moyenne	177 - 98	1037 - 195	704 - 157

TABLE 6.8 – Moyenne de l'inéquité (avant H_Δ - après H_Δ) selon les stratégies de recherche.

6.5.3 Résultats globaux

La meilleure configuration du modèle $PPC_{||U,\Delta}$, obtenue avec la stratégie LW-LN, conduit à des résultats moins bons que ceux obtenus par M2P (*cf.* Table 6.9 et Table 6.10). En ce qui concerne la faisabilité des instances, $PPC_{||U,\Delta}$ permet d'obtenir au mieux 481 solution complètes, contre 544 pour M2P. Cette différence est d'autant plus marquée que l'instance est chargée ($I_v = 1\,000$), ce qui confirme le constat du Chapitre 5 selon lequel le paramètre I_v est fortement corrélé à la difficulté des instances. Du point de vue de l'inéquité entre les employés, $PPC_{||U,\Delta}$ conduit à une inéquité moyenne d'environ deux heures, contre moins d'une heure pour M2P. De manière similaire à M2P, les solutions trouvées sont d'autant plus équitables que l'instance est de petite taille ($I_t = 100$).

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	42/47	11/21	106/122
	I	28	35	72	35
	A	97,50	97,78	96,71	97,05
	T	145	155	167	156
200	C	59/60	50/55	22/35	131/150
	I	34	35	42	36
	A	99,00	98,60	97,72	97,93
	T	166	138	156	154
300	C	58/58	53/58	42/56	153/172
	I	40	38	46	41
	A	99,67	99,19	98,76	98,94
	T	191	186	174	184
400	C	59/59	55/59	40/56	154/174
	I	47	44	51	47
	A	99,75	99,70	99,01	99,17
	T	236	202	196	211
Total	C	229/231	200/219	115/168	**544/618**
	I	38	38	49	**40**
	A	98,28	98,47	97,68	**97,90**
	T	184	170	173	**176**

TABLE 6.9 – Meilleurs résultats M2P (5 min)

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	31/47	4/21	88/122
	I	89	101	62	92
	A	97,43	97,83	95,57	96,42
	T	137	117	95	117
200	C	59/60	44/55	8/35	111/150
	I	114	116	148	118
	A	99,00	98,72	97,37	97,70
	T	58	84	78	74
300	C	58/58	51/58	30/56	139/172
	I	122	121	115	120
	A	99,67	99,22	98,74	98,89
	T	30	36	30	32
400	C	59/59	52/59	32/56	143/174
	I	132	114	138	127
	A	99,75	99,66	98,96	99,14
	T	28	35	31	31
Total	C	229/231	178/219	74/168	**481/618**
	I	115	114	126	**116**
	A	98,19	98,50	97,28	**97,64**
	T	63	68	59	**63**

TABLE 6.10 – Résultats $PPC_{||U,\Delta}$ (LW-LN, 5 min)

6.5.4 Combinaison des stratégies de recherche

L'une des conclusions du Chapitre 5 mettait en avant l'importance de diversifier les stratégies de recherche pour obtenir de meilleurs résultats. Pour cela, la méthode en deux phases applique régulièrement chacune des variantes de la seconde phase et poursuit la recherche avec la variante la plus efficace. Un moyen de transposer cette stratégie au modèle $PPC_{||U,\Delta}$ consiste à appliquer successivement plusieurs stratégies de recherche en répartissant équitablement le temps de calcul disponible. Ainsi, au lieu de résoudre le problème avec une seule stratégie de recherche et un temps limite L, il est possible de résoudre le problème k fois, avec k stratégies différentes disposant chacune d'un temps de calcul maximal L/k. L'idée sous-jacente est de bénéficier d'une éventuelle complémentarité entre les différentes stratégies utilisées. Bien que les différentes résolutions ne profitent pas les unes des autres, une telle approche peut donner de très bons résultats. Par exemple, une stratégie similaire a récemment été mise en place par l'équipe en charge du développement du solveur Choco [88] dans le cadre de la compétition MiniZinc [157], ce qui a permis au solveur Choco de se positionner en seconde place dans la catégorie « recherche parallèle ».

Nous comparons à présent les résultats du modèle $PPC_{||U,\Delta}$ pour deux configurations différentes. La première configuration correspond à une utilisation classique du modèle $PPC_{||U,\Delta}$, *i.e.* une résolution de cinq minutes avec la stratégie LW-LN, en appliquant la descente locale sur chaque solution (*cf.* Table 6.10). La seconde configuration correspond à une résolution séquentielle des différentes stratégies proposées pour le modèle $PPC_{||U,\Delta}$, *i.e.* les huit stratégies de recherche proposées sont utilisées successivement, avec un temps limite de 37,5 secondes, de manière à ce que le temps limite cumulé soit de cinq minutes. Les résultats obtenus soulignent clairement l'intérêt de ce mode de résolution (*cf.* Table 6.11). Nous obtenons en effet 532 solutions complètes, avec une inéquité moyenne de 105 minutes, contre 481 solutions complètes avec une inéquité moyenne de 116 minutes avec uniquement LW-LN. Cela valide l'intuition selon laquelle les différentes stratégies proposées peuvent

être combinées efficacement de manière à bénéficier de leur complémentarité. Ces résultats restent néanmoins moins bons que ceux obtenus avec M2P.

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	41/47	9/21	103/122
	I	59	134	296	109
200	C	59/60	48/55	19/35	126/150
	I	55	121	341	123
300	C	58/58	54/58	36/56	148/172
	I	45	108	151	94
400	C	59/59	57/59	39/56	155/174
	I	48	92	190	100
Total	C	229/231	200/219	103/168	**532/618**
	I	51	112	213	**105**

TABLE 6.11 – Meilleures solutions obtenues avec $PPC_{||U,\Delta}$ (All-All-H_Δ)

6.6 Conclusions et perspectives

Nous avons présenté dans ce chapitre une méthode intégrée fondée sur une modélisation PPC. Les contraintes légales sont donc gérées en même temps que les contraintes d'affectation, ce qui permet de créer des horaires plus fins et mieux adaptés aux tâches à couvrir. L'espace de recherche résultant étant bien plus vaste, il est a priori plus difficile à explorer. Nous proposons pour cela deux stratégies de branchement, l'une dédiée à l'obtention de solutions complètes et l'autre dédiée à l'obtention de solutions plus équitables. Nous évaluons également l'impact de plusieurs stratégies d'exploration, et montrons qu'il est possible de les combiner efficacement. Nous montrons l'intérêt de la descente locale H_Δ qui donne de très bons résultats, notamment avec MW. Au final, les meilleurs résultats obtenus sont moins bons que ceux obtenus avec M2P, ce qui illustre la difficulté d'explorer un espace de recherche plus grand, en adressant simultanément les deux niveaux de décision du problème. Des résultats préliminaires ont par ailleurs montré que l'apparition de tâches non affectées pouvait survenir relativement haut dans l'arbre de recherche. Dans ce cas, l'obtention d'une solution complète requiert de revenir très haut dans l'arbre de recherche, ce qui prend beaucoup de temps car toutes les autres feuilles de l'arbre peuvent éventuellement conduire à des solutions améliorantes du point de vue de l'équité, et doivent donc être explorée. Cela explique probablement le faible gain entre les solutions finales et initiales. Cela nous conduit à envisager une exploration plus diversifiée au moyen d'une recherche par voisinage large. De cette manière, les choix réalisés au haut de l'arbre de recherche peuvent rapidement être remis en cause, facilitant ainsi l'obtention de solutions complètes.

Dans ce chapitre, nous nous sommes concentrés sur une modélisation reposant sur l'utilisation de variables ensemblistes. Il serait intéressant de chercher à développer d'autres modélisations, voire de les combiner entre elles, ce qui conduirait alors à une modélisation redondante du problème. Ce type d'approche est souvent efficace lorsque les contraintes utilisées peuvent alors filtrer de manière complémentaire. Par ailleurs, les stratégies de recherche proposées dans ce chapitre se concentrent chacune sur une idée bien précise. Les travaux présentés dans ce chapitre ont été publiés dans Computers & Operations Research [133].

7

Une recherche à voisinage large

Nous nous intéressons ici à la résolution du SDPTSP||U,Δ via une méthode de recherche à voisinage large, plus connue sous le nom de LNS *(Large Neighborhood Search)*. Étant donnés les résultats obtenus par le modèle $PPC_{||U,\Delta}$, l'utilisation d'une LNS semble en effet pertinente. Nous rappelons tout d'abord l'idée générale de ce type d'approche et introduisons quelques travaux de référence. Nous détaillons ensuite la méthode proposée qui repose sur deux types de voisinages orthogonaux. Le premier se focalise sur un sous-ensemble d'employés, impactant ainsi tout l'horizon de planification, alors que le second se concentre sur un créneau horaire particulier, impactant ainsi tous les employés. Nous proposons également de gérer l'obtention de solutions complètes ainsi que l'obtention de plannings équitables selon des modalités différentes. Nous justifions la structure globale de la LNS ainsi que son paramétrage au moyen d'une étude expérimentale et nous concluons vis-à-vis de l'intérêt de cette méthode.

Sommaire

7.1 Introduction aux recherches à voisinages larges

Une LNS est une méthode de résolution permettant d'améliorer la qualité d'une solution initiale, via un processus itératif de destruction puis de reconstruction d'une partie de la solution courante (*cf.* Algorithme 7.1) [166, 180]. Le fait de détruire une partie de la solution courante revient à créer un voisinage qui peut alors être exploré de manière à construire une solution améliorante. La phase de reconstruction a d'autant plus de chance de conduire à une solution améliorante que le voisinage est bien choisi. Par exemple, il est recommandé d'éviter les voisinages trop contraints qui conduiraient à reconstruire la solution courante. Contrairement aux recherches locales dont le voisinage est relativement restreint, les LNS permettent de travailler sur des voisinages plus vastes, offrant ainsi l'opportunité d'échapper aux optima locaux. Autrement dit, les LNS permettent de naviguer plus librement au sein de l'espace des solutions. Cette technique présente en contrepartie l'inconvénient d'être plus gourmande en temps et en capacité de calcul. À la fin de chaque itération, une solution est sélectionnée afin de commencer une nouvelle itération. De voisinages en voisinages, ce procédé permet ainsi d'explorer des zones très diverses de l'espace de recherche. Dans le cadre d'une modélisation PPC, la phase de destruction revient généralement à désinstancier toutes les variables avant d'instancier celles qui ne font pas partie du voisinage étudié. Ainsi, au début de chaque itération, un grand nombre de variables sont déjà instanciées, ce qui permet de profiter d'un filtrage plus efficace.

En ce qui concerne la sélection des variables à désinstancier, Shaw [180] propose de sélectionner les variables présentant un lien privilégié entre elles. Par exemple, dans le cadre d'un problème de tournées de véhicules, il semble particulièrement intéressant de désinstancier les variables définissant une même route, ainsi que celles représentant des clients proches géographiquement. De manière analogue, dans le cadre d'un problème d'ordonnancement de projet avec contraintes de capacité, il semble naturel de considérer des activités concomitantes ou liées entre elles par des contraintes de précédences [163]. Il faut cependant veiller à ne pas construire les voisinages de manière trop déterministe, sous peine de sélectionner régulièrement les mêmes variables. Pour éviter cela, Shaw propose d'incorporer une dose de hasard lors de cette phase de sélection, mais il est également possible d'introduire une liste tabou, ou de sélectionner les variables à désinstancier de manière complètement aléatoire, en affectant à chaque variable une probabilité d'être désinstanciée [99].

Algorithme 7.1: Processus général de résolution d'une LNS

Données :
S_i, S_c, S_t, S_b : solutions initiale, courante, temporaire et meilleure
$\mathcal{O}_d, \mathcal{O}_r$: ensemble des opérateurs de destruction et de reconstruction
D_c, R_c : opérateurs courants de destruction et de reconstruction
C_a, C_s : critères d'arrêt et de sélection de la nouvelle solution

```
1  S_i ← resolutionInitiale()
2  (S_b, S_c) ← (S_i, S_i)                    //Initialisation des solutions
3  tant que (C_a non satisfait) faire
4      D_c ← selection(O_d)                   //Choix d'un opérateur de destruction
5      S_c ← destructionPartielle(S_c, D_c)   //Création du voisinage
6      R_c ← selection(O_r)                   //Choix d'un opérateur de reconstruction
7      S_t ← reparation(S_c, R_c)             //Construction d'une solution
8      si (S_t meilleure que S_b) alors
9          S_b ← S_t                          //Mise à jour de la meilleure solution
10     S_c ← selection(S_t, S_c, C_s)         //Sélection de la nouvelle solution courante
```

En plus du procédé de sélection des variables à désinstancier, il faut également déterminer la portée du voisinage, *i.e.* le nombre de variables qui seront désinstanciées. Plus la portée est grande, plus le voisinage peut être complexe à explorer efficacement. En revanche, plus la portée est petite, plus il est probable de reconstruire la solution courante à l'identique. Il s'agit donc de trouver un bon compromis entre l'intérêt du voisinage et la difficulté de son exploration. Par exemple, Shaw [180] propose d'incrémenter le nombre de variables à désinstancier suite à un nombre donné d'itérations successives non améliorantes. Ainsi, les améliorations simples peuvent être réalisées rapidement grâce à de petits voisinages, alors que les améliorations plus complexes, requérant de travailler sur des voisinages plus grands seront abordées en fin de résolution.

Après qu'une partie de la solution courante a été détruite, on obtient un espace de recherche qui peut ensuite être exploré partiellement ou complètement de manière à trouver, si possible, une solution améliorante. Plusieurs stratégies sont alors envisageables, la plus simple étant d'explorer tout le voisinage au moyen d'une recherche arborescente. L'avantage de cette stratégie est de garantir l'obtention de la meilleure solution locale. Cependant, lorsque la portée du voisinage augmente, chaque exploration peut prendre alors beaucoup de temps, ce qui n'est pas forcément efficace. Il est par conséquent fréquent de n'explorer qu'une partie du voisinage. Il est possible par exemple d'utiliser une LDS (Limited Discrepancy Search) afin d'explorer uniquement les zones les plus intéressantes de l'espace de recherche [180]. Il est également envisageable de limiter le nombre de backtracks ou de se limiter à la première solution trouvée. De manière générale, l'utilisation d'un modèle de PPC (ou de PLNE) est généralement associée à une exploration partielle du voisinage, car cela permet de mieux contrôler le temps accordé à chaque étape [68, 165].

L'étude de la littérature montre que les méthodes de type LNS sont très efficaces, notamment dans le cadre d'une résolution PPC. Cela peut s'expliquer par le fait que la PPC est bien adaptée lorsqu'il s'agit de trouver une solution, mais qu'elle n'est pas spécialement adaptée au parcours de l'espace de recherche dans le but d'améliorer la qualité de la solution trouvée. D'autre part, les algorithmes de filtrage sont généralement plus efficaces en bas de l'arbre de recherche, lorsque les domaines des variables sont très réduits, contrairement au haut de l'arbre où les domaines des variables sont bien plus grands. L'utilisation d'une LNS permet justement de rester dans le bas de l'arbre de recherche, tout en explorant différentes portions de l'espace de recherche et en se concentrant uniquement sur les portions les plus prometteuses. Ce procédé permet ainsi de bénéficier des mécanismes de filtrage, propagation et backtrack de la PPC, qui facilitent l'obtention d'une solution, tout en évitant le parcours d'une part importante de l'espace de recherche.

7.2 Structure et fonctionnement de la LNS proposée

Puisque le SDPTSP||U,Δ comprend à la fois un problème d'affectation de tâches et un problème de planification de personnel, il semble naturel de proposer au moins un opérateur de destruction adapté à chacun de ces problèmes. En ce qui concerne le problème d'affectation de tâches, il semble relativement naturel de définir des voisinages portant sur des créneaux horaires, afin de permettre des échanges de tâches d'un employé à un autre. En ce qui concerne le problème de planification de personnel, il semble plus naturel de définir des voisinages portant sur des employés, de manière à redéfinir leurs horaires de travail. Nous proposons par conséquent deux opérateurs de destruction. Le premier, fondé sur le temps, est noté T_{op} alors que le second, fondé sur le personnel, est noté P_{op} (*cf.* Section 7.2.1). À chaque itération, l'un de ces opérateurs est sélectionné pour détruire une partie de la solution courante. L'espace de recherche ainsi obtenu est alors exploré de manière à trouver une solution, puis nous utilisons la descente locale H_{Δ} pour réduire l'inéquité des solutions complètes ainsi que l'inéquité de la meilleure solution globale. Remarquons qu'il serait également

envisageable d'utiliser H_Δ sur toutes les solutions. Ce point précis est abordé à la Section 7.3.5. Le fonctionnement général de cette LNS est résumé à la Figure 7.1. Nous présentons par la suite les opérateurs de destruction T_{op} et P_{op} plus en détail, puis nous expliquons et justifions la stratégie d'exploration permettant de reconstruire une solution. Pour finir, nous donnons le fonctionnement détaillé de la LNS.

7.2.1 Opérateurs de destruction

Nous proposons dans cette section deux opérateurs de destruction visant à accroître le nombre de tâches affectées tout en maximisant l'équité parmi les employés. Puisqu'une solution est entièrement définie par la liste des affectations tâche-employé, le processus de destruction se réduit ici à supprimer une partie de cette liste. La sélection des affectations à supprimer doit permettre de gagner un maximum de flexibilité, de manière à permettre l'obtention de solutions améliorantes, mais avec un minimum d'affectations supprimées, de manière à bénéficier d'un filtrage efficace tout en limitant la taille du voisinage. Pour cela, il est important de prendre en compte la structure du problème.

Un opérateur de destruction fondé sur le temps : T_{op}

L'opérateur de destruction T_{op}, détaillé à l'Algorithme 7.2, permet de désaffecter un ensemble de tâches positionnées sur un créneau horaire donné, ce qui offre l'opportunité de répartir ces tâches de manière plus appropriée. Cet opérateur sélectionne tout d'abord une tâche non affectée, notée t_i, puis supprime l'affectation des tâches temporellement proche de t_i, dans la limite de la portée définie. Le créneau horaire correspondant aux tâches désinstanciées de cette manière n'est pas forcément centré autour de t_i, ce qui permet d'obtenir plusieurs espaces de recherche différents autour d'une même tâche. Lorsque toutes les tâches sont affectées, T_{op} commence par supprimer une affectation aléatoirement. Dans tous les cas, T_{op} supprime toutes les affectations de l'employé fictif (dernière ligne de l'Algorithme 7.2).

L'intérêt de l'opérateur T_{op} est illustré à la Figure 7.2. Sur cet exemple, la solution courante comprend 3 employés, notés n_1 à n_3 et 11 tâches dont une tâche non affectée, notée t (*cf.* Figure 7.2a). Chaque employé est affecté à au moins une tâche concomitante à t, ce qui fait que t ne peut être directement affectée à un employé. L'opérateur T_{op} permet de supprimer les affectations des tâches proches de t, comme par exemple les tâches t_1 à t_6. Puisque la LNS repose sur le modèle $PPC_{||U,\Delta}$, les horaires de travail des employés sont déduits à partir de l'affectation des tâches. Autrement dit, l'opérateur de destruction T_{op} permet également de redonner une certaine flexibilité aux horaires des employés. À partir de ce voisinage, il est possible d'obtenir une meilleure affectation des tâches, comme cela est illustré par la Figure 7.2b. Cet opérateur peut également permettre de réduire l'inéquité entre les employés.

Pour résumer, l'opérateur T_{op} permet de désaffecter un ensemble de tâches positionnées sur une plage horaire donnée, mais il ne permet pas de modifier les horaires de travail des employés en profondeur. Par exemple, il peut être difficile, voire impossible, de remplacer une journée de repos par une journée de travail, en raison des horaires de travail fixés avant et après la plage horaire sélectionnée. Pour les mêmes raisons, un horaire de travail de nuit peut être difficile à remplacer par un horaire de travail d'après-midi ou de matin. Autrement dit, l'opérateur T_{op} ne suffit pas à couvrir l'ensemble des modifications potentiellement nécessaires à l'amélioration de la solution courante.

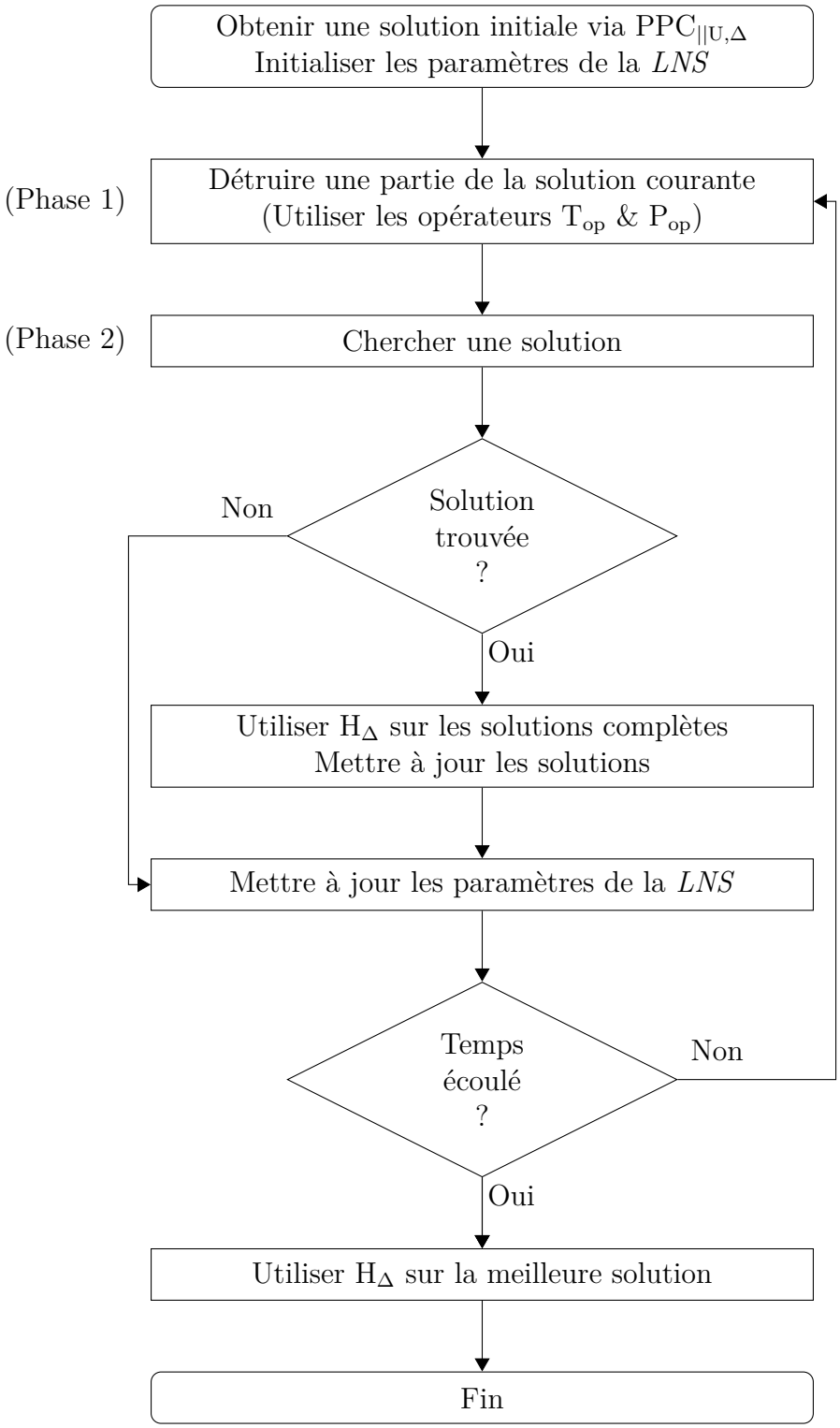
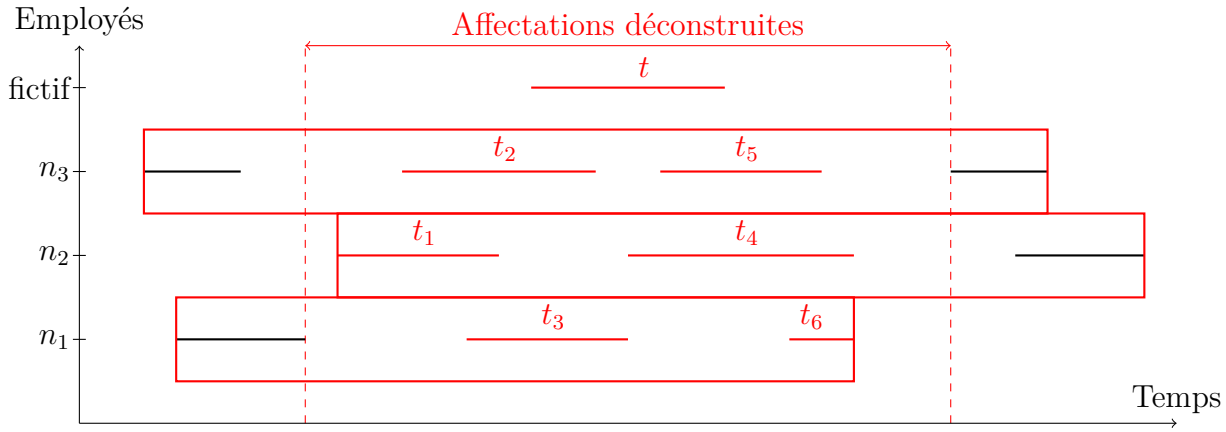


FIGURE 7.1 – Logigramme simplifié de la *LNS*.

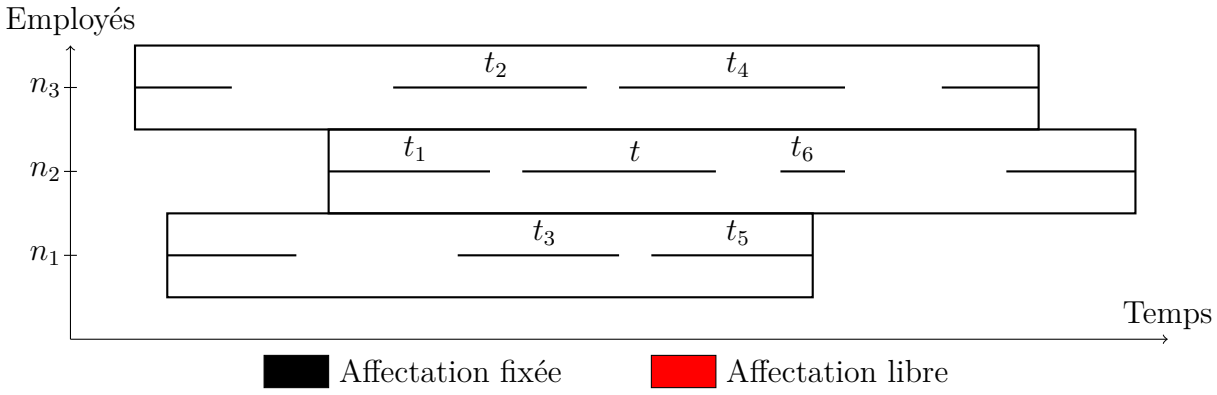
Algorithme 7.2: T_{op} : création d'un voisinage sur la dimension temps.

Données :
$\mathcal{T}$, l'ensemble des tâches
$P_{T_{op}}$, la portée actuelle du voisinage T_{op}

```
1 t ← selectionnerTâche(T)          //Si possible non affectée
2 supprimerAffectation(t, n)
3 p+ ← nombreAleatoire(1, P_Top)    //Tirage aléatoire entre 1 et P_Top
4 p− ← P_Top − p+
5 supprimerAffectationsAmont(t, p+)
6 supprimerAffectationsAval(t, p−)
7 supprimerToutesLesAffectationsFictives()
```



(a) Voisinage de la solution courante (3 employés et 11 tâches dont 1 non affectée)



(b) Solution espérée, toutes les tâches sont affectées

FIGURE 7.2 – Utilisation de l'opérateur de destruction T_{op}

Un opérateur de destruction fondé sur le personnel : P_{op}

Pour modifier en profondeur les horaires de travail des employés, il est nécessaire de modifier les affectations sur plusieurs jours, voire sur toute la semaine. Dans ce cas, il ne semble pas raisonnable de considérer l'ensemble des employés, car cela conduirait à un voisinage trop grand. Autrement dit, il s'agit de choisir un sous-ensemble d'employés et de supprimer toutes les tâches qui leur sont affectées. Pour cela, nous proposons l'opérateur P_{op}, dont le fonctionnement détaillé est donné à l'Algorithme 7.3. Notons également que P_{op} supprime toutes les affectations de l'employé fictif (dernière ligne de l'Algorithme 7.3).

L'intérêt de l'opérateur P_{op} est illustré sur un exemple à 13 tâches et 3 employés (*cf.* Figure 7.3). Sur cet exemple, la solution courante comprend 3 employés, notés n_1 à n_3 et 13 tâches dont une non affectée, notée t (*cf.* Figure 7.3a). L'affectation actuelle est telle que les employés sont tous en repos durant la plage horaire de réalisation de t. Autrement dit, il n'est pas possible d'affecter la tâche t aux employés sans modifier les horaires de travail d'au moins un employé. L'opérateur P_{op} permet de supprimer toutes les affectations des tâches sur un sous-ensemble d'employés, comme par exemple les employés n_1 et n_2, ce qui conduit à supprimer l'affectation des tâches t_1 à t_8. À partir de ce voisinage, il est possible d'obtenir une meilleure affectation des tâches, comme cela est illustré par la Figure 7.3b. Cet opérateur peut également permettre de réduire l'inéquité entre les employés.

Pour résumer, l'opérateur P_{op} permet de modifier les horaires de travail de quelques employés, en désaffectant toutes les tâches d'un sous-ensemble d'employés. De cette manière, les opérateurs P_{op} et T_{op} permettent de couvrir l'ensemble des modifications potentiellement nécessaires à l'amélioration de la solution courante.

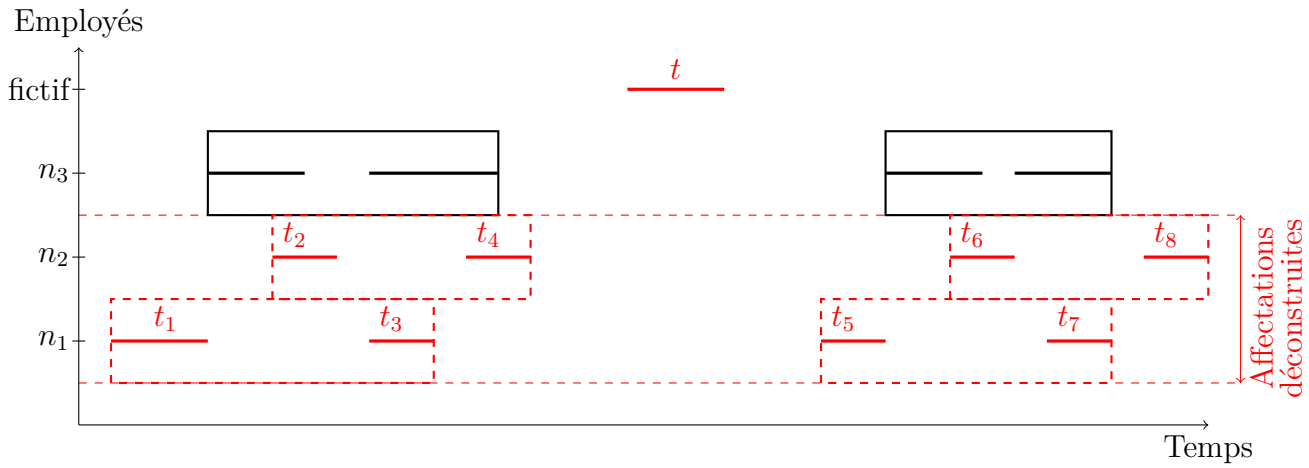
Algorithme 7.3: P_{op} : création d'un voisinage sur la dimension personnel.

Données :
$\mathcal{T}$, l'ensemble des tâches
$\mathcal{N}$, l'ensemble des employés
$\mathcal{W}$, un sous-ensemble d'employés
$P_{P_{op}}$, la portée actuelle du voisinage P_{op}

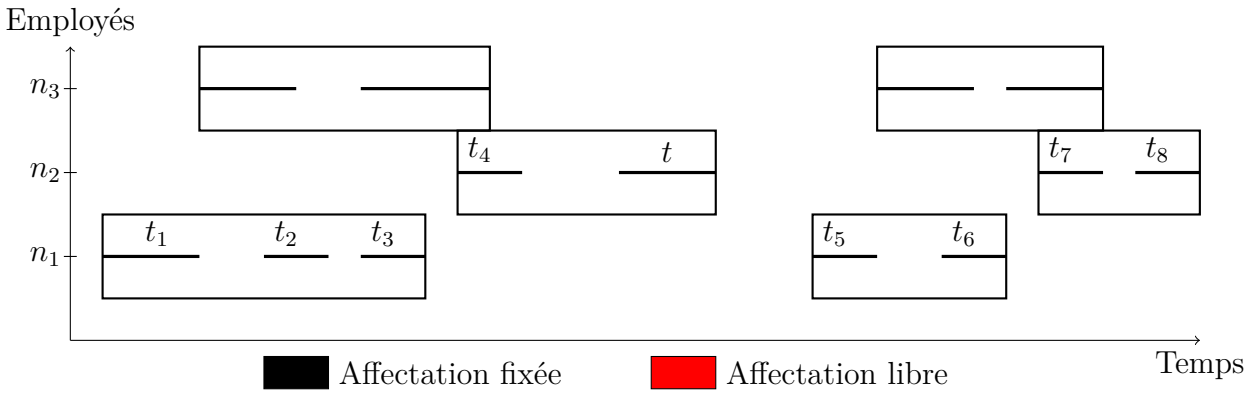
```
1 W ←selectionerEmployées(N, P_Pop)
2 pour (w ∈ W) faire
3     supprimerToutesLesAffectations(w)
4 supprimerToutesLesAffectationsFictives()
```

7.2.2 Exploration du voisinage

Les solutions partielles obtenues grâce aux opérateurs de destruction T_{op} et P_{op} sont ensuite reconstruites au moyen du modèle $PPC_{||U,\Delta}$. Étant donné que les espaces de recherche obtenus via T_{op} et P_{op} peuvent être relativement grands, nous en limitons l'exploration en nous restreignant à la première solution atteinte. Bien que le processus de destruction de la solution courante repose sur une part importante de hasard, il est possible d'obtenir des espaces de recherche très similaires, voire identiques, ce qui pose la question de la stratégie de recherche. Pour éviter de reconstruire systématiquement la même solution, il est possible d'utiliser une stratégie de recherche aléatoire ou adaptative.



(a) Voisinage de la solution courante (3 employés et 13 tâches dont 1 non affectée)



(b) Solution espérée, toutes les tâches sont affectées

FIGURE 7.3 – Utilisation de l'opérateur de destruction P_{op}

Stratégie de recherche

Les stratégies présentées à la Section 6.4, p. 100, ne sont ni aléatoire ni adaptative. Par conséquent, pour un état donné des variables, ces stratégies conduisent systématiquement à la même solution, ce qui ne semble pas adapté au mode d'exploration envisagé. Les résultats expérimentaux présentés à la Section 6.5.1, p. 102, montrent qu'une stratégie consistant à affecter prioritairement les tâches réalisables par le moins d'employés est très efficace pour obtenir des solutions complètes. Nous proposons par conséquent de conserver l'idée générale de cette stratégie tout en utilisant une stratégie adaptative. Pour cela nous introduisons les variables entières $Employe_t$ qui donnent pour toute tâche $t \in \mathcal{E}$ l'employé affecté à cette tâche. Des contraintes de liaison entre les variables $Employe$ et $Tasks$ permettent alors de garantir la cohérence entre ces deux ensembles de variables (7.1). Ces variables nous permettent de nous focaliser directement sur les tâches. Par exemple, le domaine de la variable $Employe_t$ correspond aux employés pouvant être affectés à la tâche t. Autrement dit, la stratégie consistant à brancher sur la variable $Employe$ ayant le plus petit domaine s'apparente à la stratégie d'exploration LN (*cf.* Section 6.4, p. 100).

$$\forall t \in \mathcal{E}, \forall n \in \mathcal{N}^+, \quad t \in Tasks_n \Leftrightarrow Employe_t = n \qquad \text{(chanelingSetInt)} \qquad (7.1)$$

Pour bénéficier d'une stratégie adaptative reprenant l'idée de la stratégie d'exploration LN, nous optons pour la stratégie de branchement `dom/wdeg` qui sélectionne en priorité les variables présentant le plus petit ratio entre le domaine de la variable et le nombre de contraintes liées à cette variable et ayant entraîné un backtrack [30]. De plus, nous renforçons cette heuristique de branchement au moyen de l'heuristique `last-conflict` qui sélectionne en priorité les variables à l'origine du dernier backtrack [136]. De cette manière, plus le domaine d'une variable $Employe$ est petit et plus son affectation semble difficile, plus cette variable sera sélectionnée rapidement. Ainsi, au fil des résolutions, le poids de chaque contrainte évolue, ce qui modifie le branchement sur les variables. En ce qui concerne la stratégie d'exploration, nous optons pour la stratégie `min value` qui consiste simplement à choisir la plus petite valeur du domaine. Le fait de choisir systématiquement la plus petite valeur du domaine conduit à construire en priorité le planning des premiers employés. Dans une certaine mesure, cette stratégie s'apparente par conséquent au branchement MW qui vise à conserver un maximum de flexibilité vis-à-vis de la construction des horaires de travail (*cf.* Section 6.4). Au final, nous obtenons une stratégie adaptative qui reprend l'idée de stratégies MW et LN qui sont efficaces pour trouver des solutions complètes.

7.2.3 Gestion de l'objectif

L'objectif du problème est de trouver une solution avec un maximum de tâches affectées et, parmi toutes les solutions équivalentes sur cet aspect, de trouver une solution qui minimise l'inéquité. Pour atteindre cet objectif, nous avons vu à la Section 6.3, p. 100 qu'il était possible de considérer simultanément les objectifs U et Δ de manière lexicographique. Comme nous l'avons signalé à la Section 6.6, p. 106, cela conduit à une exploration peu efficace de l'espace de recherche. Par conséquent, nous proposons ici une stratégie visant à trouver une solution équivalente vis-à-vis de l'objectif U, sans contraindre l'objectif Δ. L'intérêt de cette stratégie est de faciliter l'obtention de solutions au moins aussi bonnes du point de vue du nombre de tâches non affectées, ce qui permet de « naviguer » plus facilement parmi l'ensemble des solutions et ainsi éviter les optima locaux. Cela s'apparente en quelques sortes à un mécanisme de diversification, ce qui est très classique dans le cadre des méta-heuristiques. Par ailleurs, la descente locale H_Δ étant très efficace, l'inéquité des solutions trouvées peut être améliorée dans un second temps grâce à cet algorithme. Comme nous l'avons signalé à la Section 4.2.3, p. 66, H_Δ ne conserve par l'ordre des solutions. Dans ce contexte, il peut être plus efficace de chercher à obtenir beaucoup de solutions équivalentes du point de vue du nombre de tâches non affectées, au lieu de quelques solutions améliorantes du point de vue de l'inéquité.

7.2.4 Fonctionnement détaillé de la LNS

Le fonctionnement détaillé de la LNS est donné à l'Algorithme 7.4. Notons tout d'abord qu'à chaque itération, l'opérateur de destruction utilisé est sélectionné aléatoirement. Ensuite, lors de l'exploration du voisinage, nous nous contentons de chercher une solution au moins aussi bonne du point de vue du nombre de tâches non assignées. La descente locale H_{Δ} est ensuite utilisée sur chaque solution complète pour réduire l'inéquité. Lorsqu'une solution améliorante est trouvée, la portée de l'opérateur de destruction courant est réduite. En revanche, au bout d'un certain nombre d'itérations non améliorantes, la portée de l'opérateur est augmentée. Dans tous les cas, la portée ne dépasse pas ses valeurs extrêmes. Par ailleurs, notons qu'à chaque nouvelle itération, nous construisons le voisinage à partir de la meilleure solution globale. En effet, puisqu'à la fois la destruction de la solution courante ainsi que l'exploration de l'espace de recherche reposent sur une part importante de hasard et puisque nous acceptons les solutions non améliorantes, il ne semble pas nécessaire de construire le voisinage sur la meilleure solution locale pour diversifier la recherche.

Algorithme 7.4: Structure générale de la LNS proposée

Données :
S_i, S_c, S_b : solutions initiale, courante et meilleure
$P_o, P_o^i, \underline{P_o}, \overline{P_o}$: portées courante, initiale, minimale et maximale de l'opérateur o
o, P_o^+ : opérateur de destruction courant et modification de portée de l'opérateur o
I_c, I_m : nombres courant et maximal d'itérations non améliorantes

$S_i \leftarrow$ resolutionInitialePPC()
$(S_b, S_c, I_c, P_{T_{op}}, P_{P_{op}}) \leftarrow (S_i, S_i, 0, P^i_{T_{op}}, P^i_{P_{op}})$ *//Solutions et paramètres initiaux*
tant que *(temps de calcul non écoulé)* **faire**
  $o \leftarrow$ selectionAléatoire(T_{op}, P_{op})
  $S_c \leftarrow$ destructionPartielle(S_b, P_o, o) *//Création du voisinage*
  $S_c \leftarrow$ reparationPPC(S_c) *//Sélection de la première solution*
  $S_c \leftarrow \mathrm{H}_{\Delta}(S_c)$
  si (S_c *meilleure que* S_b)) **alors**
    $I_c \leftarrow 0$
    $P_o \leftarrow \max(P_o - P_o^+, \underline{P_o})$ *//Réduction de la portée*
    $S_b \leftarrow S_c$ *//Mise à jour de la meilleure solution*
  sinon
    $I_c \leftarrow I_c + 1$
    si $(I_c = I_m)$ **alors**
      $P_o \leftarrow \min(P_o + P_o^+, \overline{P_o})$ *//Accroissement de la portée*
      $I_c \leftarrow 0$

7.3 Résultats expérimentaux

Nous présentons dans cette section les résultats de la LNS[1]. Pour rappel, la qualité des résultats est évaluée selon les indicateurs :

C : le nombre total d'instances pour lesquelles la méthode trouve une solution complète.

[1] Les algorithmes sont implémentés en JAVA et nous utilisons Choco 3-1-1 [62] pour implémenter $\mathrm{PPC}_{||\mathrm{U},\Delta}$. Les tests ont été réalisés sur un Intel Core i3-540 (3.06 GHz & 8G RAM)

I : l'inéquité moyenne des solutions complètes (en minutes).
A : le pourcentage moyen de tâches affectées pour les solutions partielles uniquement.
T : le temps moyen d'obtention des meilleures solutions (en secondes).

7.3.1 Résultats globaux

Globalement, la LNS conduit à de meilleurs résultats que ceux obtenus par M2P (*cf.* Tables 7.1 et 7.2). En ce qui concerne la faisabilité des instances, la LNS permet d'obtenir 581 solutions complètes, contre 544 pour M2P. Par ailleurs, certains jeux d'instances difficiles à résoudre (I_t= 100 et I_v= 1 000) sont bien mieux résolus avec la LNS qu'avec M2P. Du point de vue de l'inéquité entre les employés, la LNS conduit à une inéquité moyenne d'environ 27 minutes, contre 40 pour M2P. De manière similaire à M2P, les solutions trouvées sont d'autant plus équitables que l'instance est de petite taille. En ce qui concerne le nombre moyen de tâches affectées pour les solutions partielles, la LNS donne également des résultats légèrement meilleurs que M2P (98,12% contre 97,90%). Le temps moyen d'obtention de la meilleure solution est globalement plus petit avec la LNS qu'avec M2P, mais cela tient essentiellement aux instances de petites tailles. Ces résultats valident l'intérêt d'une approche visant à renforcer les capacités d'exploration de $PPC_{||U,\Delta}$.

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	54/54	46/47	20/21	120/122
	I	17	14	17	16
	A	97,67	98,07	97,10	97,38
	T	144	117	128	127
200	C	59/60	53/55	29/35	141/150
	I	27	24	26	26
	A	99,00	98,71	98,90	98,13
	T	73	175	162	162
300	C	58/58	54/58	47/56	159/172
	I	33	26	29	29
	A	99,67	99,33	98,90	99,10
	T	180	170	171	172
400	C	59/59	58/59	44/56	161/174
	I	34	31	38	34
	A	99,75	99,75	99,27	99,34
	T	278	289	195	209
Total	C	230/231	211/219	139/168	**581/618**
	I	28	24	30	**27**
	A	98,41	98,60	97,95	**98,12**
	T	158	154	155	**155**

TABLE 7.1 – Résultats de la LNS (5 min)

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	42/47	11/21	106/122
	I	28	35	72	35
	A	97,50	97,78	96,71	97,05
	T	145	155	167	156
200	C	59/60	50/55	22/35	131/150
	I	34	35	42	36
	A	99,00	98,60	97,72	97,93
	T	166	138	156	154
300	C	58/58	53/58	42/56	153/172
	I	40	38	46	41
	A	99,67	99,19	98,76	98,94
	T	191	186	174	184
400	C	59/59	55/59	40/56	154/174
	I	47	44	51	47
	A	99,75	99,70	99,01	99,17
	T	236	202	196	211
Total	C	229/231	200/219	115/168	**544/618**
	I	38	38	49	**40**
	A	98,28	98,47	97,68	**97,90**
	T	184	170	173	**176**

TABLE 7.2 – Résultats M2P (5 min)

7.3.2 Qualité de la solution initiale

Nous comparons à présent la qualité de la solution initiale obtenue avec la stratégie de recherche `dom/deg + last conflict-min value` par rapport aux solutions initiales obtenues avec LW-LN (*cf.* Tables 7.3 et 7.4). Les résultats expérimentaux montrent que la stratégie de recherche utilisée avec la LNS permet d'obtenir une trentaine de solutions complètes supplémentaires par rapport à LW-LN. Autrement dit, le fait de se focaliser sur les tâches difficiles à affecter se révèle très efficace. En

contrepartie, l'inéquité moyenne est en revanche plus élevée, mais cela ne se répercute pas sur les solutions finales, comme nous l'avons vu à la Section 7.3.1.

Puisque les solutions initiales sont en moyenne très satisfaisantes, il semble important de mesurer la robustesse de l'approche en l'initialisant avec une solution moins bonne. Nous comparons par conséquent les résultats de la LNS au bout de 5 minutes en faisant varier la stratégie de recherche utilisée pour initialiser la méthode. Dans tous les cas, dès que la solution initiale est trouvée, nous employons à nouveau la stratégie de recherche `dom/deg + last conflict-min value`. Nous avons testé les stratégies MW-LN et LW-LN qui donnent respectivement 467 et 481 solutions complètes (contre 513 pour la stratégie générale) avec une équité d'environ 240 et 140 minutes (contre 339 pour la stratégie générale). Au final, nous obtenons des résultats presque identiques à ceux de la Table 7.1, ce qui montre que l'efficacité de la méthode ne tient pas tant à la qualité de la solution initiale qu'à sa capacité à l'améliorer. Plus précisément, la stratégie MW-LN permet d'obtenir 579 solutions complètes avec une équité d'environ 27 minutes, alors que la stratégie LW-LN permet d'obtenir 582 solutions complètes avec une équité moyenne de 29 minutes. Autrement dit, cette dernière configuration permet d'améliorer très légèrement les résultats obtenus par la LNS.

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	31/47	4/21	88/122
	I	129	154	108	137
200	C	59/60	44/55	8/35	111/150
	I	133	144	164	140
300	C	58/58	51/58	30/56	139/172
	I	132	132	139	134
400	C	59/59	52/59	32/56	143/174
	I	141	129	162	141
Total	C	229/231	178/219	74/168	**481/618**
	I	134	138	150	**138**

TABLE 7.3 – Solutions initiales $PPC_{||U,\Delta}$ (LW-LN-H_Δ)

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	52/54	37/47	5/21	94/122
	I	280	487	684	383
200	C	59/60	48/55	12/35	119/150
	I	260	368	416	319
300	C	58/58	51/58	36/56	145/172
	I	271	308	471	334
400	C	59/59	56/59	40/56	155/174
	I	225	325	497	331
Total	C	228/231	192/219	93/168	**513/618**
	I	258	362	486	**339**

TABLE 7.4 – Solutions initiales LNS (`dom/deg + last conflict-min value`-H_Δ)

7.3.3 Intérêt des opérateurs de destruction

Pour mesurer l'efficacité et la complémentarité des opérateurs T_{op} et P_{op}, nous comparons à présent les résultats obtenus avec la LNS lorsqu'un seul des opérateurs est utilisé (*cf.* Tables 7.5 et 7.6). Par rapport à la solution initiale (*cf.* Tables 7.4) les résultats obtenus avec un seul des opérateurs sont bien meilleurs. En ce qui concerne la réalisabilité des instances, l'opérateur T_{op} permet d'obtenir 41 nouvelles solutions complètes, contre 53 pour P_{op}. De cette manière chaque opérateur, pris séparément, conduit à de meilleurs résultats que M2P. Par ailleurs, le fait que la combinaison de ces deux opérateurs conduise à davantage de solutions complètes souligne leur complémentarité. En ce qui concerne l'inéquité des solutions complètes, chaque opérateur permet de passer en dessous de 30 minutes, ce qui est très satisfaisant. Remarquons également que les opérateurs T_{op} et P_{op} obtiennent des résultats assez différents selon les différents jeux d'instances. Par exemple, T_{op} obtient la plus grande inéquité pour les instances de petites tailles mais fortement chargées (I_t= 100 et I_v= 1 000) alors que la tendance s'inverse pour P_{op}. En revanche, le nombre moyen de tâches affectées ainsi que le temps moyen d'obtention des meilleures solutions sont presque identiques. Notons pour finir que le temps moyen d'obtention des meilleures solutions tourne pour chaque opérateur autour de 230

secondes contre 140 lorsque les deux opérateurs sont utilisés. Autrement dit, l'utilisation conjointe des deux opérateurs permet de converger plus rapidement vers de bonnes solutions.

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	44/47	14/21	111/122
	I	14	16	106	27
	A	97,86	98,13	96,76	97,19
	T	265	220	71	185
200	C	59/60	50/55	19/35	128/150
	I	20	17	25	20
	A	99,00	98,75	98,11	98,25
	T	295	251	99	215
300	C	58/58	54/58	44/56	156/172
	I	24	19	44	28
	A	99,67	99,22	98,92	99,06
	T	290	270	221	260
400	C	59/59	57/59	43/56	159/174
	I	31	24	48	33
	A	99,75	99,75	99,18	99,29
	T	295	285	221	267
Total	C	229/231	205/219	120/168	**554/618**
	I	23	19	50	**27**
	A	98,46	98,63	97,85	**98,06**
	T	286	257	153	**232**

TABLE 7.5 – Résultats de la LNS avec l'opérateur T_{op} uniquement (5 min)

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	45/47	15/21	113/122
	I	10	10	16	11
	A	97,86	98,07	97,04	97,36
	T	265	225	77	189
200	C	59/60	51/55	25/35	135/150
	I	18	13	19	16
	A	99,00	98,83	98,06	98,23
	T	295	255	128	226
300	C	58/58	54/58	46/56	158/172
	I	27	22	20	23
	A	99,67	99,33	98,93	99,11
	T	290	272	232	265
400	C	59/59	57/59	44/56	160/174
	I	35	28	29	31
	A	99,75	99,75	99,19	99,30
	T	294	285	229	269
Total	C	229/231	207/219	130/168	**566/618**
	I	23	19	22	**21**
	A	98,46	98,66	97,92	**98,12**
	T	286	259	167	**237**

TABLE 7.6 – Résultats de la LNS avec l'opérateur P_{op} uniquement (5 min)

En ce qui concerne la portée des voisinages, des résultats préliminaires nous ont conduits à adopter un paramétrage dépendant de la taille de l'instance. Plus précisément, la portée de l'opérateur T_{op} (respectivement P_{op}) varie entre 5 et 30% du nombre de tâches (respectivement 5 et 30% du nombre d'employés, avec un minimum de deux employés) avec un accroissement de portée d'un quart toutes les 50 itérations non améliorantes. En ce qui concerne la variation de la portée, il serait également envisageable d'incrémenter la portée des opérateurs ce qui simplifie légèrement le paramétrage du mécanisme général. Des résultats préliminaires ont montré que les résultats obtenus avec une variation incrémentale de la portée sont similaires à ceux obtenus avec une variation par « bonds ». Nous avons néanmoins conservé le mécanisme de variation par « bonds », qui nous a semblé plus robuste. Bien entendu, de nombreux paramétrages alternatifs sont envisageables et il semble difficile de déterminer la meilleure configuration.

Pour évaluer l'intérêt du mécanisme de modification de la portée des voisinages, nous le comparons à une autre configuration où la portée des voisinages est fixée à sa valeur minimale et n'est jamais augmentée (*cf.* Table B.13). Globalement, les résultats obtenus sont moins bons à plusieurs points de vue, ce qui valide l'intérêt d'un mécanisme d'accroissement de la taille des voisinages. Plus précisément, nous perdons 13 instances complètes, dont 8 sur les petites instances très chargées (I_t= 100, I_v= 1 000). De plus, le temps d'obtention moyen des meilleures solutions passe de 150 à 200 secondes. L'équité moyenne reste la même, mais cela cache en réalité des différences assez marquées. Par exemple, sur les instances peu chargées (I_v= 600), nous gagnons en moyenne 4 minutes, alors que sur les instances les plus chargées (I_v= 1 000), nous perdons en moyenne 12 minutes, avec une

I_t	I_v			
	600	800	1 000	Total
100	616-17	619-14	524-17	602-16
200	691-27	638-24	467-26	625-26
300	749-33	705-26	630-29	699-29
400	803-34	726-31	657-38	735-34
Total	717-28	675-24	590-30	**671-27**

TABLE 7.7 – Meilleure équité moyenne (avant H_Δ - après H_Δ) des solutions complètes

perte maximale de près d'une heure pour les instances les petites instances très chargées. Pour résumer, ce mécanisme n'est pas essentiel sur les instances les plus faciles, mais il permet d'améliorer significativement les résultats sur les instances les petites instances très chargées.

7.3.4 Gestion de l'objectif

Nous évaluons à présent la pertinence de la stratégie de gestion de l'objectif, présentée à la Section 7.2.3, p. 115. Une approche plus classique consisterait à minimiser strictement le nombre de tâches non assignées ainsi que l'inéquité. Cette configuration permet d'obtenir 580 solutions complètes avec une équité moyenne de 80 minutes. Autrement dit, le fait de minimiser le nombre de solutions complètes ne permet pas d'en trouver davantage alors que le fait de minimiser strictement l'inéquité conduit à des solutions globalement moins bonnes, par manque de diversité des solutions obtenues. En revanche, les solutions de petites tailles (I_t= 100) sont alors mieux résolues, puisque l'inéquité moyenne tourne autour de 10 minutes (contre 16 dans le cas contraire). Nous en concluons qu'il est globalement plus efficace de chercher à obtenir beaucoup de solutions pour les améliorer ensuite grâce à la descente locale H_Δ, plutôt que de chercher directement des solutions strictement améliorantes. Pour confirmer cette intuition, nous avons comparé d'autres configurations en variant les différentes combinaisons possibles de politique de minimisation. Lorsqu'on accepte les solutions avec plus de tâches non assignées, on réduit alors le nombre de solutions complètes à 542. Autrement dit, il est important de chercher à faire au moins aussi bien vis-à-vis du nombre de tâches non assignées, mais il n'est pas efficace de chercher à faire mieux. En revanche, en ce qui concerne l'inéquité des solutions, il est systématiquement plus efficace d'accepter les solutions non améliorantes.

7.3.5 Impact de H_Δ

Nous avons vu à la Section 7.3.1 que la LNS permettait d'obtenir de très bons résultats en terme d'équité. Puisque ces résultats correspondent à l'utilisation successive de la LNS à proprement parler, ainsi que de la descente locale H_Δ, nous cherchons à présent à mesurer l'impact de H_Δ sur l'inéquité des solutions complètes trouvées avec la LNS. La Table 7.7 donne la valeur moyenne des solutions complètes les plus équitables avant et après la descente locale H_Δ. Contrairement à M2P, la LNS ne cherche pas à améliorer directement l'équité, mais à obtenir beaucoup de solutions différentes qui sont ensuite améliorées grâce à H_Δ. Par conséquent, les meilleures solutions avant H_Δ sont très inéquitables (environ 11 heures). Ce résultat est cohérent, puisque ni la stratégie de recherche, ni les opérateurs ni visent particulièrement à réduire l'inéquité. En revanche, l'inéquité moyenne des solutions après H_Δ est très bonne (environ 30 minutes). Ces résultats valident l'idée selon laquelle la descente locale H_Δ peut être utilisée efficacement pour trouver de bonnes solutions. En revanche, contrairement à M2P, H_Δ devient alors un élément indispensable de la méthode.

Nous avons vu à la Section 7.2, p. 109, que la descente locale H_Δ était utilisée sur la meilleure

solution globale et sur chaque solution complète. Pour trouver des solutions plus équitables, il pourrait être avantageux d'utiliser H_Δ sur chaque solution. Cela pourrait en revanche réduire le nombre de solutions complètes, comme nous l'avons observé avec M2P (*cf.* Section 5.4.3, p. 92). Les résultats expérimentaux sont en fait très proches de ceux obtenus en appliquant H_Δ sur chaque solution complète uniquement. Cette configuration conduit à perdre une solution complète, passant ainsi à 580, alors que l'inéquité moyenne tourne autour de 28 minutes contre 27 avec la deuxième configuration. Il est intéressant de remarquer que le temps moyen d'obtention de la meilleure solution est cependant légèrement plus court lorsque H_Δ est systématiquement utilisée. Nous en déduisons que le fait d'utiliser H_Δ sur chaque solution permet soit de faire converger la méthode vers des solutions équitables, soit de diversifier les solutions explorées. Autrement dit, H_Δ permet également de naviguer dans l'espace de recherche de manière efficace.

7.3.6 Comparaison des méthodes

I_t	Méthode	I_v			
		600	800	1 000	Total
100	PLNE	60	60	59	179
	PPC	58	50	17	125
	LNS	60	60	59	179
	M2P	57	48	29	134
200	PLNE	60	48	35	143
	PPC	60	49	27	136
	LNS	60	60	60	180
	M2P	60	54	32	146
300	PLNE	56	31	20	107
	PPC	60	58	41	159
	LNS	60	60	60	180
	M2P	60	57	49	166
400	PLNE				
	PPC	60	59	40	159
	LNS	60	60	60	180
	M2P	60	57	43	160
Total	PLNE	176	139	114	429
	PPC	238	216	126	580
	LNS	240	240	239	719
	M2P	237	216	153	606

TABLE 7.8 – Nombre de meilleurs résultats pour U

I_t	Méthode	I_v			
		600	800	1 000	Total
100	PLNE			1	1
	PPC		2	3	5
	LNS	56	54	41	151
	M2P	4	4	16	24
200	PLNE	1			1
	PPC	7		4	11
	LNS	47	52	50	149
	M2P	9	9	7	25
300	PLNE				
	PPC	28	6	3	37
	LNS	26	50	46	122
	M2P	8	5	11	24
400	PLNE				
	PPC	40	9	6	55
	LNS	20	42	47	109
	M2P	1	13	9	23
Total	PLNE	1		1	2
	PPC	75	17	16	108
	LNS	149	198	184	531
	M2P	22	31	43	96

TABLE 7.9 – Nombre de meilleurs résultats pour (U,Δ)

Nous comparons à présent les résultats de la LNS par rapport aux autres méthodes : M2P et $PPC_{||U,\Delta}$ (version combinée des stratégies) sur 5 minutes et $PLNE_{||U}$ sur 10 heures. Les résultats agrégés sont donnés aux Tables 7.8 et 7.9 alors que les résultats détaillés sont donnés à l'Annexe B.4, p. 178. Globalement les résultats sont très satisfaisants puisque la LNS obtient les meilleurs résultats sur 531 instances, contre respectivement 108, 96 et 2 pour les approches $PPC_{||U,\Delta}$, M2Pet $PLNE_{||U}$ (*cf.* Table 7.9). En ce qui concerne le nombre de tâches non affectées uniquement, la LNS obtient les meilleurs résultats sur 718 instances, contre respectivement 580, 606 et 429 pour les approches $PPC_{||U,\Delta}$, M2P et $PLNE_{||U}$ (*cf.* Table 7.9). Autrement dit, lorsqu'il s'agit d'affecter un maximum de tâches, la LNS est clairement la méthode la plus efficace. Les résultats sont un peu plus mitigés en ce qui concerne l'obtention d'une solution équitable, puisque $PPC_{||U,\Delta}$ donne de bons résultats sur

les instances de grandes tailles, même si la LNS reste globalement l'approche la plus efficace, ce qui s'explique par la différence de stratégie de recherche.

7.4 Conclusions et perspectives

Nous avons présenté dans ce chapitre une méthode de recherche par voisinage large (LNS). Cette méthode exploite plusieurs caractéristiques du problème. Tout d'abord, deux opérateurs de voisinages complémentaires permettent d'améliorer rapidement et fortement la qualité de la solution initiale. De plus, une stratégie de recherche plus adaptée permet de trouver de meilleures solutions initiales. Par ailleurs, nous exploitons certaines propriétés de la descente locale H_Δ et appliquons une politique de gestion de l'objectif plus fine, ce qui permet de réduire fortement l'inéquité des solutions. Dans l'ensemble, cette méthode est plus rapide et plus efficace que M2P, à la fois vis-à-vis du nombre de solutions complètes et vis-à-vis de l'inéquité. Par ailleurs, le temps d'obtention des meilleures solutions est en moyenne plus faible qu'avec M2P.

Dans ce chapitre, nous nous sommes concentrés sur les mécanismes de destruction d'une partie de la solution courante et nous avons opté pour une reconstruction fondée sur le modèle $PPC_{||U,\Delta}$ avec une heuristique de branchement adaptative. Puisqu'en moyenne les meilleurs solutions sont trouvées bien avant le temps limite de résolution, il serait envisageable d'utiliser un port-folio de stratégies de recherche pour explorer chaque voisinage selon différentes stratégies, ce qui s'est révélé très efficace au Chapitre 6. Notons d'ailleurs que le fait d'initialiser la LNS avec la stratégie LW-LN a permis d'améliorer légèrement les résultats, ce qui tend à confirmer que l'utilisation de plusieurs stratégies de reconstruction pourrait améliorer les résultats globaux. Néanmoins, il faudrait alors trouver un bon compromis entre le nombre de stratégies utilisées et le temps de calcul requis pour chaque stratégie supplémentaire. Pour cela, une première possibilité serait de faire évoluer la LNS actuelle vers une *ALNS* (*Adaptative Large Neighborhood Search* [166]). L'idée de cette approche est d'utiliser un mécanisme adaptatif de sélection des opérateurs de destruction et de reconstruction. Autrement dit, plus un opérateur est efficace, plus il est fréquemment utilisé. Cela permet de réduire l'impact des opérateurs les moins performants. Il serait également possible de mettre en place un mécanisme de diversification similaire à celui employé pour la deuxième phase de M2P, *i.e.* le problème d'affectation de tâches à horaires fixés. De cette manière, au bout d'une certain nombre d'itérations, nous pourrions tester à nouveau toutes les stratégies de recherche, puis poursuivre la résolution avec la meilleure stratégie. Pour finir, il serait également intéressant de tester d'autres opérateurs de destructions, fondés davantage sur une logique « solveur » que sur une logique métier. Par exemple, les opérateurs génériques proposés par Perron *et al.* [164], fondé sur la propagation des contraintes, et par Prud'homme *et al.* [169], fondé sur les explications peuvent constituer de bonnes pistes.

8

MAESTRO, un logiciel de gestion et d'optimisation

Nous présentons dans ce chapitre le logiciel d'aide à la décision que nous avons développé pour Biotrial. Nous commençons par un bref état des lieux, de manière à mieux cerner le contexte technique du projet. Par la suite, nous détaillons les différentes fonctionnalités implémentées et montrons plus particulièrement comment utiliser ces fonctionnalités pour simplifier la gestion du planning du personnel. Nous décrivons également les grandes étapes du projet ainsi que ses différents acteurs.

Sommaire

8.1 Données opérationnelles et processus actuels

Nous présentons ici un bref état des lieux du processus général de planification du personnel de Biotrial. Plus précisément, nous expliquons le rôle des différents acteurs ainsi que la chronologie des

grandes étapes du processus de planification. Nous présentons également les différents documents de gestion utilisés par Biotrial.

8.1.1 Données générales

L'effectif de travail de Biotrial comprend une trentaine d'employés, ce qui est suffisamment restreint pour que les décideurs soient capables de retenir par cœur les compétences et le contrat de chaque employé. Le cas des intérimaires est en revanche plus délicat. Bien entendu, le contrat de travail des intérimaires ainsi que leurs horaires de travail sont connus dès le recrutement, mais il n'en va pas de même pour leur profil de compétence. En théorie, certains actes médicaux, jugés classiques, sont automatiquement maîtrisés par les intérimaires, mais en pratique, Biotrial constate fréquemment un manque de qualification des intérimaires ou bien des difficultés d'adaptation au rythme et à la logique de travail. Par conséquent, le profil de compétence d'un intérimaire n'est jamais connu a priori, sauf si l'intérimaire en question a déjà travaillé chez Biotrial. Dans le cas contraire, les compétences et l'efficacité de l'intérimaire sont évaluées directement sur le terrain. Autrement dit, il n'existe pas de document officiel faisant état des compétences maîtrisées par chaque employé travaillant chez Biotrial.

Biotrial réalise quotidiennement une multitude d'actes médicaux. Certains sont relativement classiques, tels que les prises de sang, de tension, ou de température, alors que d'autres sont plus compliqués, comme par exemple les tests cognitifs ou les tests de résistance à la douleur. Par ailleurs, certains actes, tels que les encéphalogrammes et électrocardiogrammes ou encore l'analyse d'échantillons, requièrent un matériel spécifique, dont l'usage nécessite une formation poussée. Tous ces actes dépendent des protocoles de tests fournis par les laboratoires pharmaceutiques. Autrement dit, chaque étude apporte son lot d'actes particuliers. Pour cette raison, il n'existe actuellement aucun document formel établissant la liste des actes cliniques réalisés par Biotrial. Cela dit, sous l'impulsion du projet MAESTRO et par souci d'homogénéisation, Biotrial est en train de composer une base des actes récurrents qui pourra être enrichie au fur et à mesure.

8.1.2 Données spécifiques à l'activité

La plus grande partie de l'activité de Biotrial est surveillée au moyen de trois types de fichiers Excel : le **planning d'activité**, le **planning du personnel** et les **plannings d'étude**. Ces documents permettent de piloter l'activité de Biotrial sur les plans tactique et opérationnel. Contrairement aux informations sur les contrats et sur les compétences, ces données sont renseignées régulièrement et sont représentées sous un format spécifique à l'activité de Biotrial.

Planning d'activité : il s'agit d'un document représentant les jours d'études des groupes pour toutes les études en cours. Ce document, élaboré par les infirmières responsables de la planification de chaque groupe de volontaires, se présente sous la forme d'un diagramme de Gantt où figurent plusieurs informations, telles que le nombre de volontaires hospitalisés et en ambulatoire ainsi que la charge de travail estimée de chaque jour. Ce document permet de piloter l'activité de Biotrial sur un plan tactique, en évaluant de manière macroscopique les capacités d'accueil de Biotrial.

Planning d'étude : il s'agit d'un document indiquant les actes à réaliser sur chaque volontaire et pour chaque jour d'étude. Ce document se présente sous la forme d'un tableau Excel où figurent la liste de tous les actes médicaux à réaliser, ainsi que leurs horaires de réalisation pour chaque volontaire. Un système de coloration permet de regrouper plusieurs actes/volontaires pour former une tâche unique, non préemptive et à affecter à un unique employé. Contrairement au planning d'activité, il existe autant de plannings d'étude que d'études en cours, soit environ une dizaine pour chaque semaine. Les plannings d'étude sont réalisés en premier lieu par l'équipe

projet correspondante, puis ajustés par les infirmières planificatrices. Ce document est très important pour Biotrial, car il sert de référence aux employés et permet également d'alimenter le planning du personnel en donnant la liste des tâches médicales à réaliser chaque semaine.

Planning du personnel : il s'agit d'un document indiquant sur un horizon d'une semaine l'activité de chaque employé au fil du temps. Ce document est réalisé manuellement par les infirmières en chef, selon un procédé incrémental et itératif. Dans un premier temps, les jours de repos ainsi que les réunions et autres indisponibilités sont ajoutés sur le planning de chaque employé, ce qui conduit à la « trame du planning », *i.e.* le planning obligatoire de chaque employé. Par la suite, cette trame est complétée en affectant des tâches aux employés, ce qui se fait de manière itérative, jusqu'à ce que toutes les tâches soient affectées et que le planning soit de bonne qualité, tout en respectant toutes les contraintes du problème. Ce processus peut éventuellement déclencher l'embauche d'intérimaires supplémentaires de manière à couvrir toutes les tâches.

Jour de cinétique : il s'agit d'un jour d'étude particulièrement chargé vis-à-vis des tâches médicales techniques, *i.e.* se déroulant au laboratoire. Ces tâches sont souvent plus longues que les actes cliniques et peuvent nécessiter un matériel précis et en quantité limité, comme une centrifugeuse par exemple. Il est donc important de visualiser le positionnement de ces journées de cinétique pour vérifier que le matériel disponible est suffisant et pour prévoir un nombre suffisant d'employés de manière à couvrir toutes les tâches.

Un exemple de planning d'activité, représentant une partie du mois de septembre 2011, est donné à la Figure 8.1. Nous pouvons y voir deux études (*Étude 1* & *Étude 2*) ainsi que plusieurs groupes (*G2a*, *G2b*, *G3a*, *etc.*). Les labels des cellules, tels que *D8**, *D9*, *D10*, *etc.* correspondent à des jours d'études, *i.e.* des jours requérant la prise en charge des volontaires et/ou la réalisation d'actes médicaux. Les jours marqués d'un astérisque correspondent à des jours de **cinétique**. Comme nous pouvons le voir, tous les jours d'étude ne sont pas forcément consécutifs, par exemple les jours d'études *FE**, correspondant aux fin d'études, surviennent ici après une période de repos d'une semaine. Par ailleurs, le total journalier du nombre de lits utilisés par les volontaires hospitalisés, ainsi que le nombre de volontaires en ambulatoire permet d'évaluer les capacités d'accueils restantes.

Un exemple de planning du personnel pour la semaine 22 est donné à la Figure 8.2. Nous pouvons y voir trois employés, ainsi que tous les jours de la semaine. L'employé 2 est indisponible lundi, mardi et dimanche, alors que l'employé 3 est indisponible samedi. Les tâches réalisées par les employés sont représentées par un label, comme par exemple *Day12 Etude1 (Gr2) PK (21h00-21h16) + Pip*, indiquant le jour, l'étude, le groupe de volontaires, l'acte médical ainsi que ses dates de début et de fin. Chaque label est associé à une couleur correspondant à une étude donnée, ce qui permet de repérer facilement la répartition des études parmi les employés. Par exemple, l'employé 1 travaille sur cinq études différentes, alors que les employés 2 et 3 ne travaillent que sur deux études. Lorsqu'un employé est affecté à plusieurs tâches du même jour, de la même étude, pour le même groupe, cette information commune n'est pas répétée, mais chaque horaire correspond bien à une tâche à part entière. Ainsi, par exemple, l'employé 2 est affecté à quatre tâches différentes le jeudi soir (19h05-19h55, 20h05-20h50, 20h55-22h20, 23h00-01h50).

Un exemple de planning d'étude avec coloration est donné à la Figure 8.3. On peut constater que le décideur a regroupé tous les actes d'administration, de manière à ce qu'ils soient réalisés par un même employé. De la même manière, les électrocardiogrammes, les pressions artérielles et les prélèvements sanguins des volontaires n°1, n°3 et n°5 ont été regroupés en un seul acte commençant à 7h30 et s'étalant sur 48 minutes.

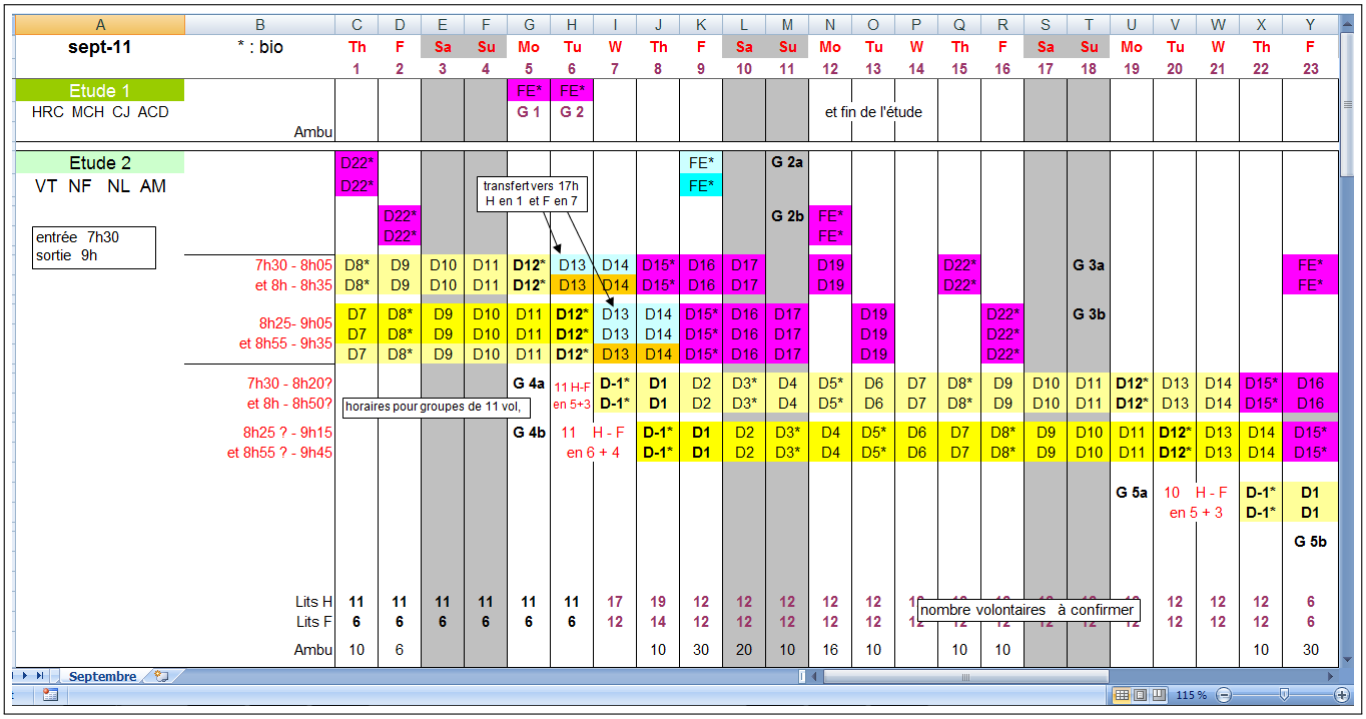


FIGURE 8.1 – Exemple d'un planning d'activité

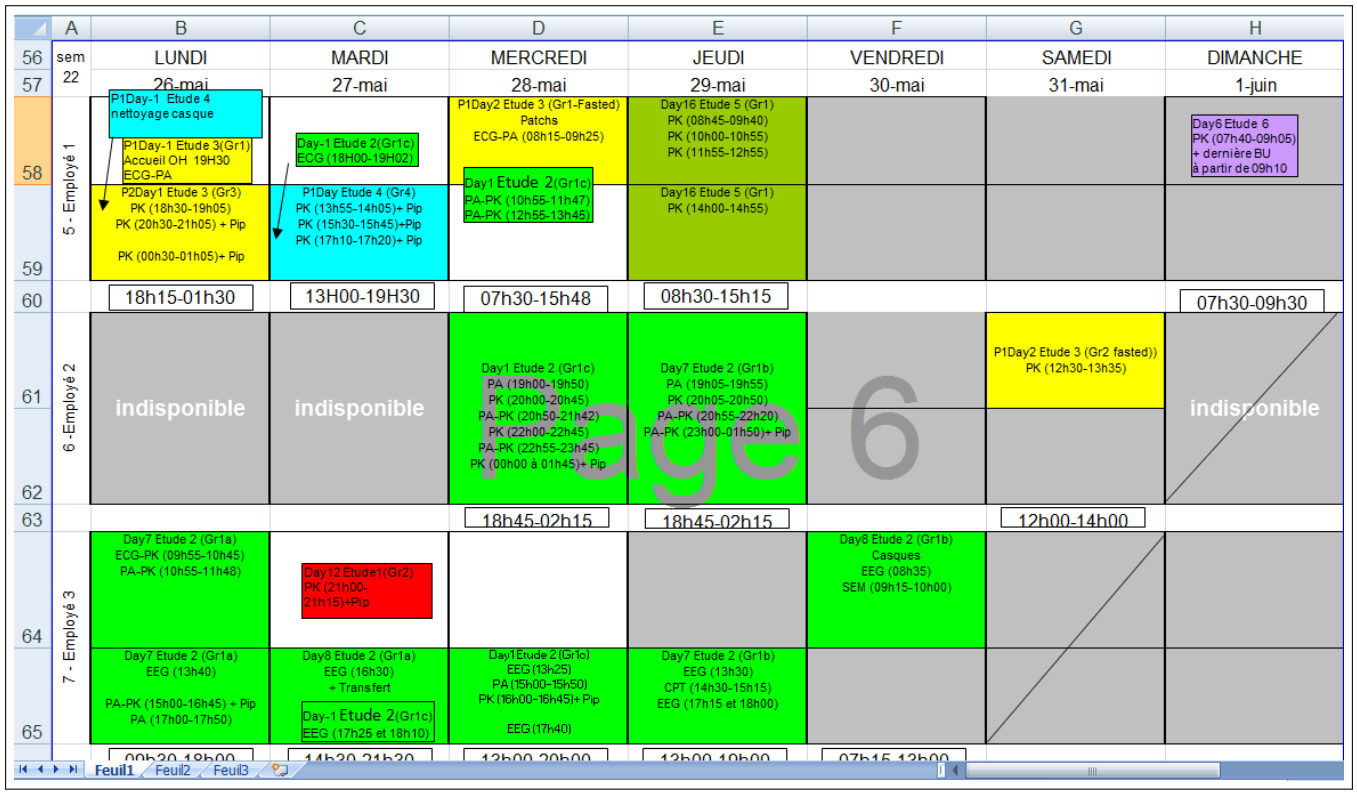


FIGURE 8.2 – Exemple d'un planning du personnel

Day1									
Temps Théor. Absolu	**Temps Théorique**	**Paramètres**	**Vol 1**	**Vol 2**	**Vol 3**	**Vol 4**	**Vol 5**	**Vol 6**
D1		Evénements indésirables et traitements	NA					
	Prédose	Pose de cathéter	NA					
01:00		Petit déjeuner (30 min)	07:00	07:05	07:10	07:15	07:20	07:25
00:30		Heure de couchée (10 min)	07:30	07:35	07:40	07:45	07:50	07:55
00:10		Electrocardiogramme - ***Visite 6***	07:50	07:55	08:00	08:05	08:10	08:15
00:08		Electrocardiogramme - ***Visite 7***	07:52	07:57	08:02	08:07	08:12	08:17
00:06		Electrocardiogramme - ***Visite 8***	07:54	07:59	08:04	08:09	08:14	08:19
00:05		Pression artérielle + Température	07:55	08:00	08:05	08:10	08:15	08:20
00:02		Prélèvement sanguin *(2 x 2ml, puis dans la glace)*	07:58	08:03	08:08	08:13	08:18	08:23
00:00	**H0**	**Administration (avec 240ml d'eau)**	08:00	08:05	08:10	08:15	08:20	08:25

FIGURE 8.3 – Exemple d'un planning d'étude

8.1.3 Processus actuel

Lorsque Biotrial reçoit une demande de test pharmaceutique, les infirmières en chef évaluent tout d'abord la faisabilité de l'étude, en fonction principalement de la capacité d'accueil des volontaires. À ce stade, l'effectif disponible n'est généralement pas pris en compte, sachant qu'un manque d'employé sera compensé au moyen d'intérimaires. Une fois que l'étude est acceptée, les infirmières en chef constituent une équipe projet (*cf.* Section 2.1.1, p. 23). Dans un premier temps, l'équipe projet étudie le protocole de tests de l'étude, de manière à en extraire la liste des actes médicaux à réaliser. Ces actes sont ensuite renseignés au sein du planning d'étude, puis regroupés en tâches médicales au moyen d'un système de coloration (*cf.* Section 2.1.2, p. 24).

Dès que tous les actes médicaux sont renseignés, le médecin de l'équipe projet peut valider le protocole de tests. À ce stade, les infirmières en chef vérifient la coloration de l'équipe projet afin d'homogénéiser les regroupements d'actes médicaux qui peuvent varier d'une équipe-projet à l'autre. En effet, certaines personnes ont tendance à sous-évaluer la difficulté du travail, et proposent alors des regroupements de tâches trop ambitieux, alors que d'autres surestiment la difficulté du travail, et proposent alors des regroupements de tâches peu efficaces. Par ailleurs, certains actes pouvant être regroupés sont parfois séparés lorsque l'activité n'est pas trop chargée, pour un plus grand confort de travail. Actuellement, aucune consigne précise ne permet de standardiser ce processus de coloration qui dépend fortement de la charge de travail et de contraintes métier difficiles à capter. Autrement dit, l'harmonisation de ce processus requiert d'une part une vue d'ensemble de l'activité, mais également une grande expertise métier. Cette tâche est par conséquent confiée aux infirmières en chef.

Une fois que le planning d'étude est finalisé, les infirmières en chef peuvent alimenter les plannings du personnel avec les tâches médicales correspondant aux études. Autrement dit, chaque semaine, les infirmières en chef consultent le planning d'activité afin de lister les jours d'études à planifier, puis pour chacun de ces jours, elles consultent les plannings d'études pour lister les tâches médicales à réaliser. Une fois que cette liste est connue, il faut alors affecter chaque tâche médicale à un employé compétent et disponible, de manière à construire des horaires de travail respectant les contraintes légales et organisationnelles (*cf.* Section 2.1.4, p. 26). Bien que tous les actes médicaux soient fixés dans le temps, les tâches attribuées aux employés correspondent elles à des regroupements d'actes médicaux, susceptibles d'être modifiés au fil de l'étude en fonction des besoins. De plus, une étude donnée peut être annulée au dernier moment pour des problèmes administratifs, ou bien en raison d'un problème de recrutement des volontaires. Par conséquent, le processus de planification manuelle est très itératif, long, complexe et stressant pour les employés et les infirmières en chef. Les premiers doivent en effet s'adapter à une activité très variable et à un rythme de travail soutenu, alors que

les secondes doivent régulièrement revoir le planning initial, dans des délais parfois très courts. De manière à faire face à ces difficultés, les infirmières en chef prennent souvent de l'avance sur le planning du personnel. Pour cela, elles évaluent grossièrement la charge de travail relative aux protocoles de tests en cours, sans attendre que l'équipe projet ait finalisé le planning d'étude. De cette manière, il devient possible de réaliser rapidement une ébauche du planning du personnel, permettant ainsi d'évaluer les besoins en intérimaires de manière à prévoir les recrutements d'intérimaires suffisamment en avance.

8.2 MAESTRO, un outil d'aide à la decision

Biotrial a récemment entrepris des travaux d'agrandissement et de restructuration visant à rapatrier l'activité du site de Rueil sur le site de Rennes, tout en augmentant la capacité d'accueil des volontaires. Ces travaux permettront de doubler le nombre de lits disponibles. Par conséquent, cette extension devrait logiquement s'accompagner d'un fort accroissement du nombre de données à prendre en compte quotidiennement au sein du service de Rennes : nombre d'employés, nombre d'études, nombre de volontaires. Étant donné que le processus de planification manuelle est actuellement long, complexe, stressant et générateur d'importants besoins en intérimaires, Biotrial souhaite mettre en place un outil informatique lui permettant de rationnaliser son activité, tout en gagnant en efficacité et en confort pour les employés. Le projet visant à mettre en place cet outil, ainsi que l'outil lui même ont été baptisés MAESTRO, pour ***M****edical* ***A****ttendanc****E*** ***S****hif****T*** ***R****ostering* ***O****ptimisation*. Nous détaillons par la suite le périmètre de l'application MAESTRO, ainsi que ses fonctionnalités, en nous intéressant notamment au moteur de planification issu des travaux de cette thèse.

8.2.1 Périmètre et structure de l'application

D'un point de vue fonctionnel, le but principal de MAESTRO est de faciliter et d'améliorer le processus de planification du personnel. Les infirmières en chef souhaitent donc que MAESTRO permette de générer des plannings automatiques, de les modifier et de les vérifier. De plus, MAESTRO doit également permettre une mise à jour simple et efficace d'un planning donné suite à une modification des données d'entrée. Pour cela, l'application doit prendre en compte de nombreuses données, telles que les contrats des employés, leurs compétences et leurs disponibilités ainsi que leur charge médicale idéale, mais également toutes les données relatives aux études, comme par exemple les équipes-projet, le profil des études, les volontaires des études et les actes médicaux des études. Certaines de ces données sont relativement bien formalisées au sein de Biotrial, alors que d'autres sont beaucoup plus informelles, ce qui nécessite par conséquent de clarifier ces données.

D'un point de vue technique, Biotrial souhaite que MAESTRO soit facilement accessible à tous les employés via un navigateur internet. En effet, comme nous avons pu le voir à la section précédente, le processus de planification du personnel est fortement lié à la planification des études qui est réalisée par les équipes-projets. Autrement dit, bien que le but principal de MAESTRO soit d'assister les infirmières en chef lors de la planification du personnel, MAESTRO doit également pouvoir être utilisé efficacement par l'ensemble du personnel. Par ailleurs, Biotrial prévoit de déployer cette application sur son antenne à New York, ce qui nécessite un effort particulier d'internationalisation.

De manière à répondre à ces besoins, nous avons proposé une application web de type client riche, associée à un moteur de planification. En ce qui concerne la gestion des données, remarquons tout d'abord que plusieurs éléments, tels que les compétences, les contrats des employés et les actes médicaux, n'étaient pas formalisés au sein de Biotrial, et par conséquent, il n'était pas possible de les charger directement. En revanche, d'autres données, telles que les plannings d'études et d'activité sont disponibles sous format Excel, comme illustré par les Figures 8.1 et 8.3. Cependant, puisque

MAESTRO est destiné à l'ensemble du personnel, nous avons estimé que les données d'entrée seraient plus stables en passant par une interface de saisie au lieu d'un système de chargement des données opérationnelles. Pour finir, le fait de gérer directement toutes les données d'entrée facilite leur mise à jour et permet d'interagir efficacement entre les plannings d'études et le planning du personnel. Par conséquent, nous avons décidé de mettre en place une base de données relationnelle regroupant toutes les informations nécessaires au fonctionnement de MAESTRO.

La version actuelle de MAESTRO comprenant six onglets principaux et deux onglets de moindre importance :

1. « Bienvenue » : informations d'ordre général.
2. « Législation » : permet de gérer les contrats de travail et par la même, les contraintes légales.
3. « Actes » : permet de maintenir à jour les données des actes médicaux (durée, intitulé).
4. « Personnel » : permet de gérer l'effectif de travail (disponibilités, contrat, compétences).
5. « Etudes » : permet de gérer les études et renseigner les plannings d'études.
6. « Gantt » : permet de consulter le planning d'activité.
7. « Planning » : permet de gérer les plannings du personnel.
8. « DevMode » : permet de lancer certains services de maintenance.

L'onglet « Bienvenue » est le point d'entrée de MAESTRO pour tous les utilisateurs. Les onglets « Législation », « Personnel » et « Gantt » sont accessibles uniquement aux infirmières en chef. Les onglets « Actes » et « Planning » sont accessibles à tous les employés en lecture, mais seules les infirmières en chef peuvent modifier les données et utiliser les fonctionnalités de planification et de modification d'un planning. Pour finir, l'onglet « Études » est accessible en lecture seule par tous les employés, les équipes-projet ont les droits de modification de leurs études, et les infirmières en chef ont tous les droits d'accès sur cet onglet. L'onglet « DevMode » est accessible uniquement au service informatique de Biotrial et aux développeurs contribuant sur le projet.

8.2.2 Détail des fonctionnalités

L'onglet « Législation » permet de créer, supprimer et modifier des contrats de travail. Rappelons que seules les infirmières en chef y ont accès. Cet onglet permet par conséquent d'alimenter les contraintes légales prises en compte pour chaque employé par le moteur de planification. Remarquons que les contraintes organisationnelles, telles que la non ubiquité des employés, n'apparaissent pas dans cet onglet, qui est exclusivement réservé aux contraintes légales. Une capture d'écran de cet onglet est donnée à la Figure 8.4.

L'onglet « Actes » permet de créer, supprimer et modifier les actes médicaux répertoriés (*cf.* Figure 8.5). Rappelons que tous les employés y ont accès en lecture seule, mais que seules les infirmières en chef peuvent modifier les données. Les actes médicaux dont la durée par volontaire est fixée sont renseignés dans la catégorie clinique, alors que les autres appartiennent à la catégorie des actes techniques. La durée des actes techniques est donc fixée au cas par cas, dans l'onglet « Etude ».

L'onglet « Personnel » permet de créer, supprimer et modifier les employés. Rappelons que seules les infirmières en chef y ont accès. Nous distinguons quatre catégories d'employés : 1) les infirmiers, qui réalisent de préférence les actes cliniques ; 2) les techniciens, qui réalisent de préférence les actes techniques ; 3) les médecins, qui valident et supervisent les études ; 4) les intérimaires. Les employés sont identifiés via leurs initiales et sont associés à un type de contrat, ainsi qu'une charge médicale idéale (*cf.* Figure 8.6). La gestion des intérimaires est presque identique à celle des autres employés. La seule différence est que chaque intérimaire est associé à des dates de disponibilités (*cf.* Figure 8.7).

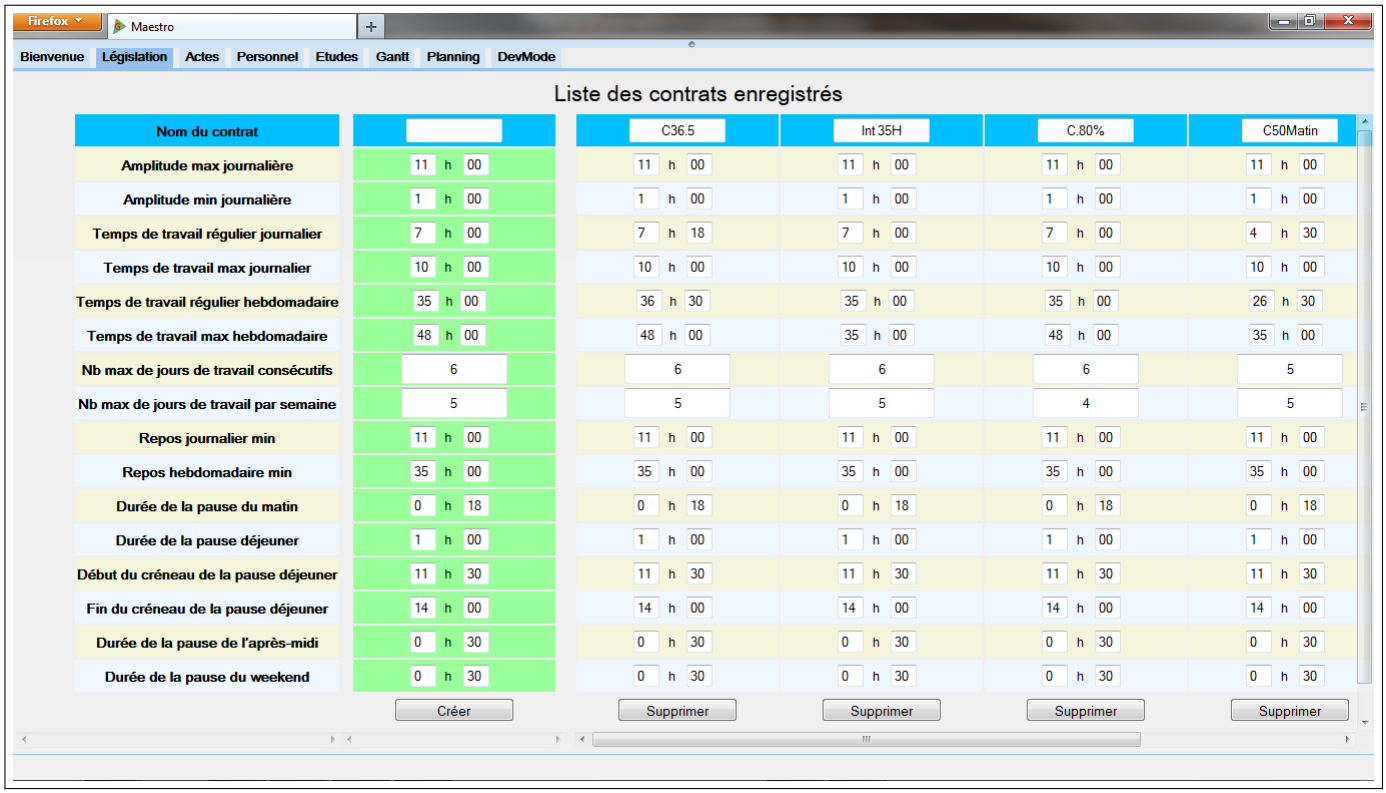


FIGURE 8.4 – Capture d'écran de l'onglet « Législation ».

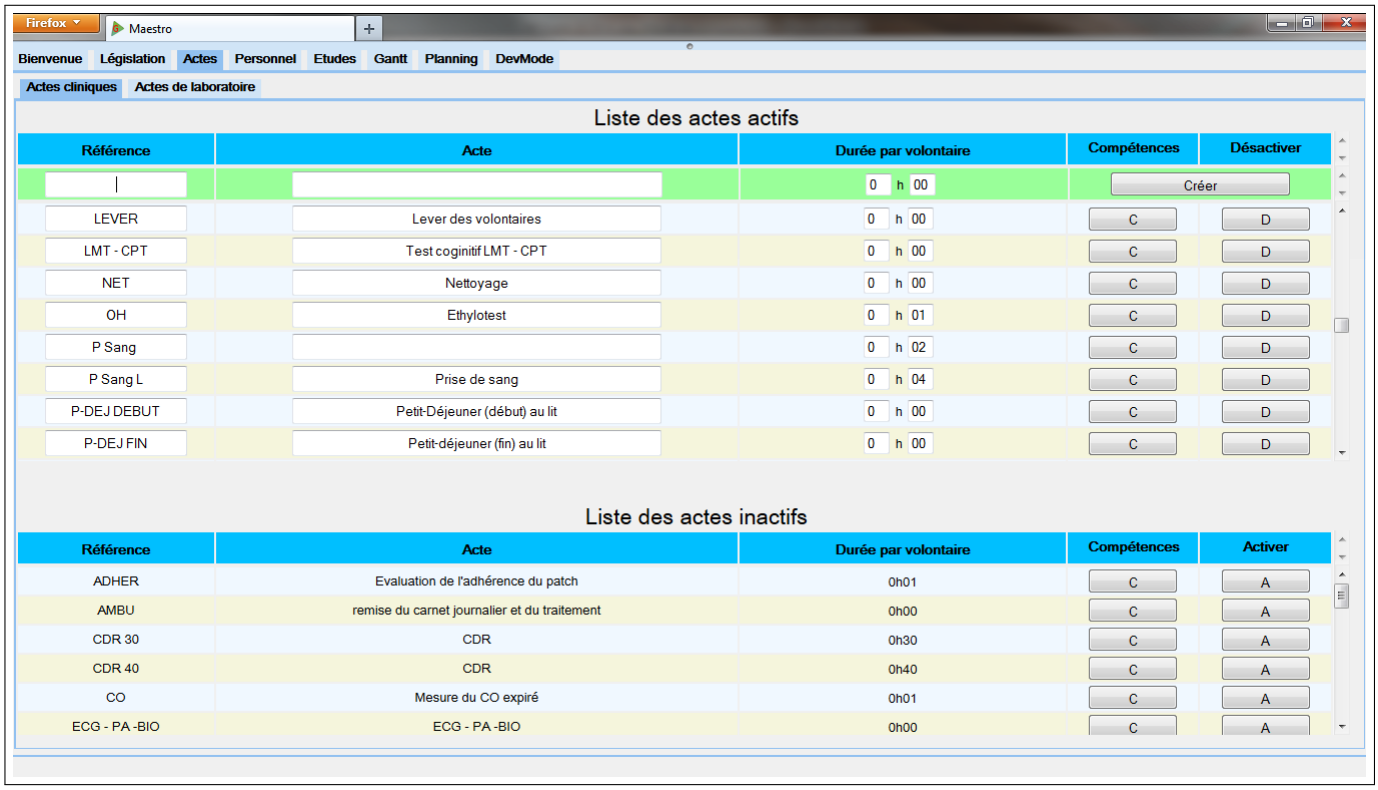


FIGURE 8.5 – Capture d'écran de l'onglet « Actes ».

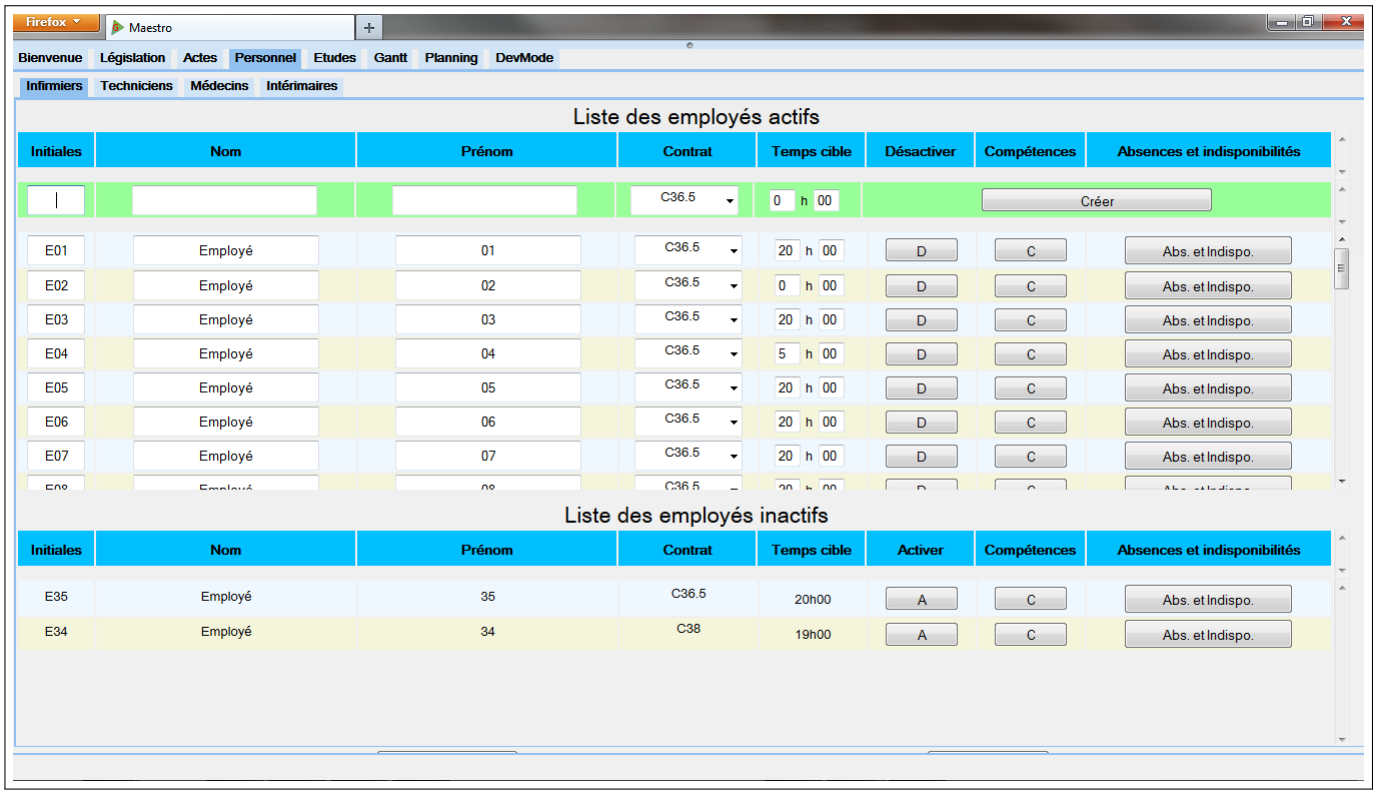


FIGURE 8.6 – Capture d'écran de l'onglet « Personnel ».

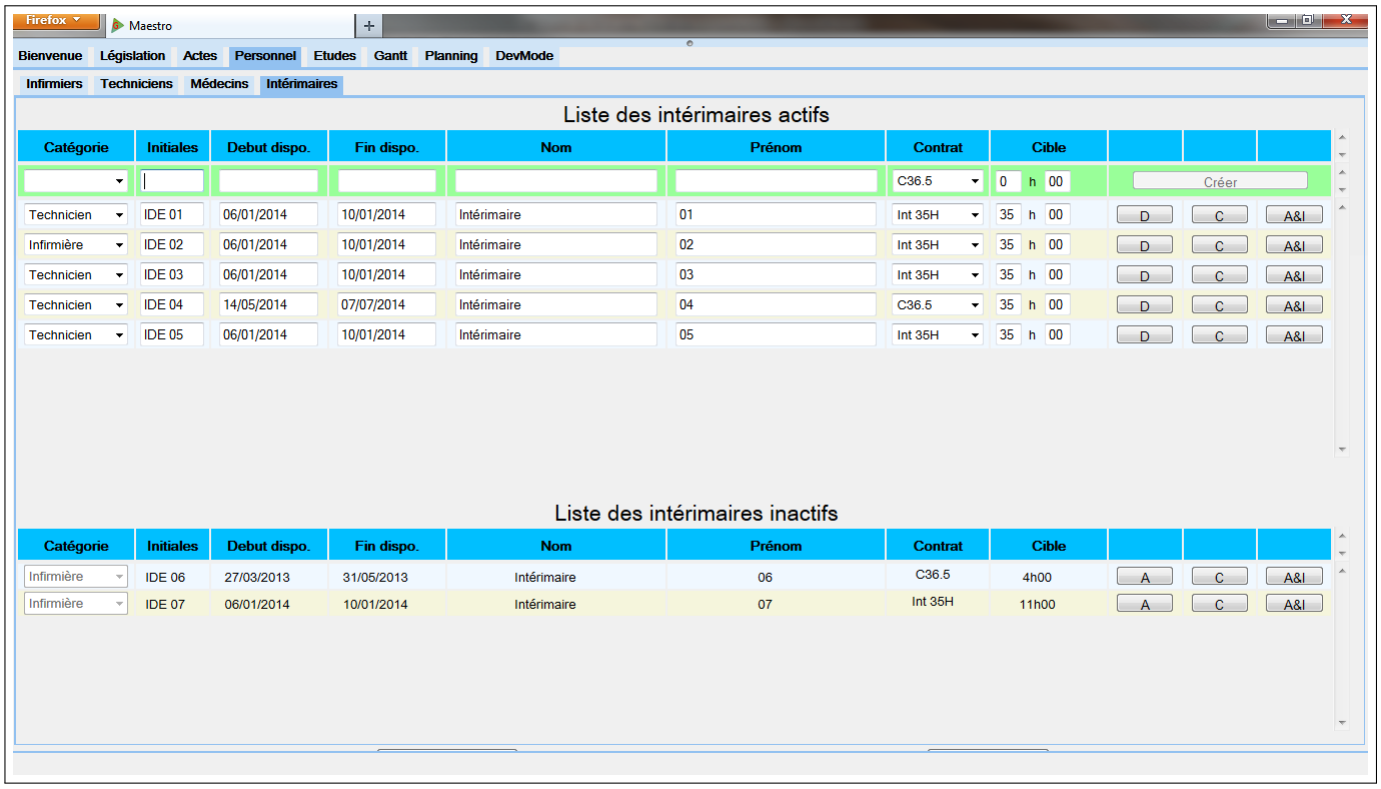


FIGURE 8.7 – Capture d'écran de l'onglet « Intérimaires ».

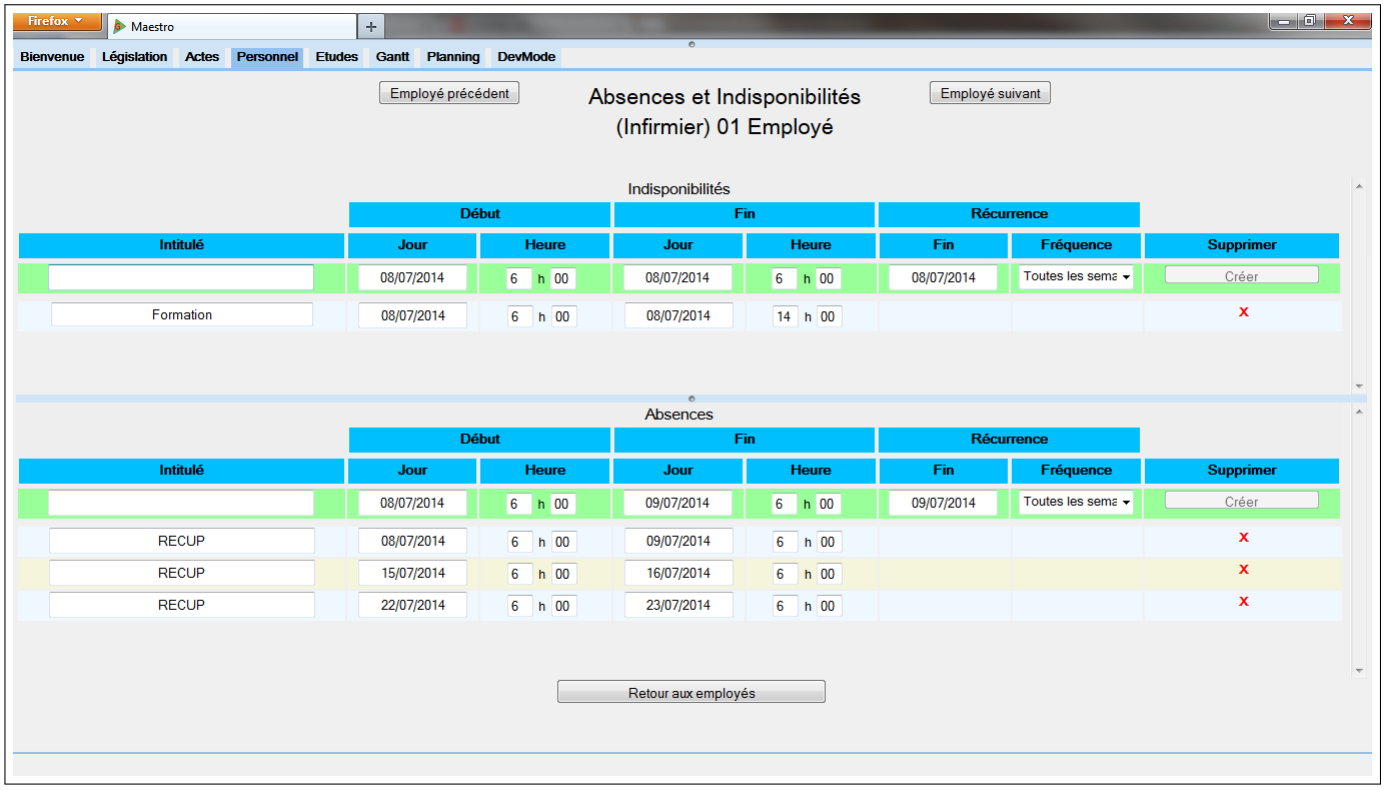


FIGURE 8.8 – Capture d'écran de l'onglet « Absences et Indisponibilités ».

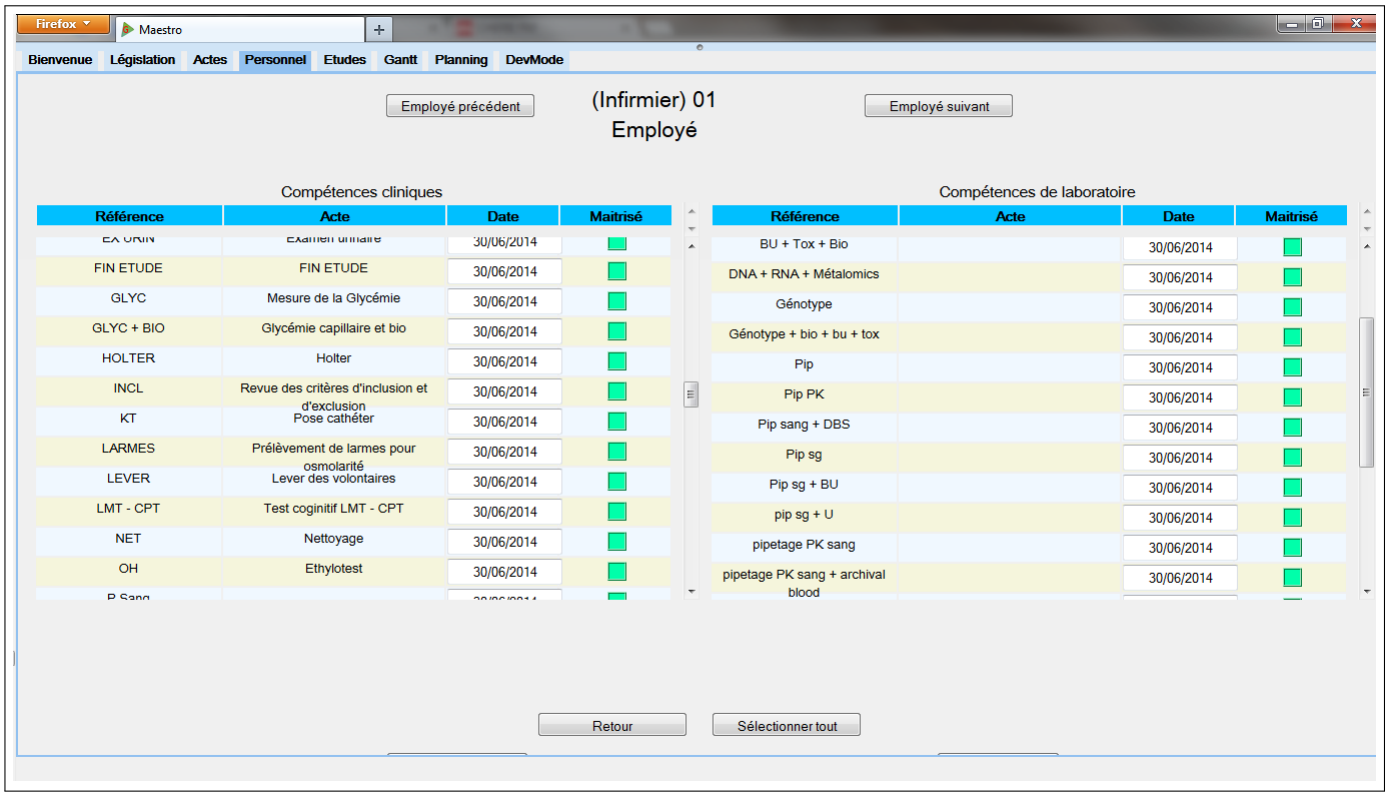


FIGURE 8.9 – Capture d'écran de l'onglet « Compétences d'un employé ».

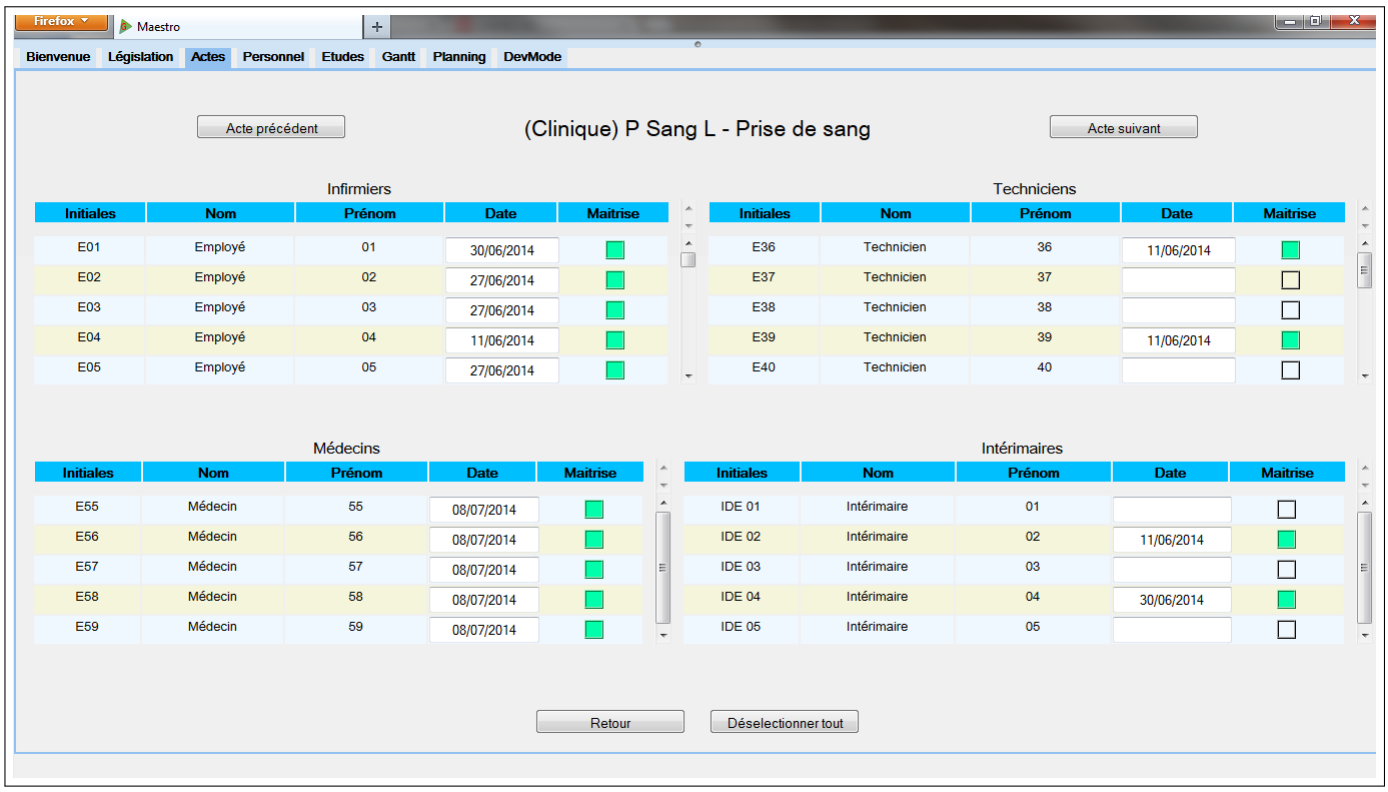


FIGURE 8.10 – Capture d'écran de l'onglet « Attribution d'une compétence aux employés ».

Cet onglet permet également d'accéder aux onglets « Absences et Indisponibilités » (*cf.* Figure 8.8), ainsi qu'à l'onglet « Compétences » (*cf.* Figure 8.9). Le premier permet de renseigner les formations, réunions et absences des employés, alors que le second permet de modifier les compétences d'un employé. Il est important de faire la différence entre les indisponibilités travaillées (indisponibilités) et les indisponibilités non travaillées (absences) car cela conditionne la légalité des plannings générés par MAESTRO. En ce qui concerne les compétences des employés, il est possible de leur associer une date de maîtrise de la compétence, ce qui permet de planifier les formations des employés en avance et de prendre en compte cette montée en compétence. Remarquons qu'un onglet similaire est également accessible depuis l'onglet « Acte » et permet d'attribuer la maîtrise d'un acte à plusieurs employés à la fois (*cf.* Figure 8.10). De cette manière, le décideur peut au choix, renseigner le profil de compétences d'un employé, ou bien modifier les employés qui maîtrisent un acte particulier.

Dans l'onglet « Études », les sous onglets « Création des études », « Profil des études » et « Tâches des études » regroupent presque toutes les fonctionnalités relatives aux études. Plus précisément, l'onglet « Création des études » permet de créer de nouvelles études et de modifier le nombre de groupe des études (*cf.* Figure 8.11). Il est également possible d'attribuer une couleur aux études de manière à identifier rapidement les tâches de cette étude dans l'onglet « Planning ». Par ailleurs, il est également possible d'attribuer les droits de gestion sur une étude particulière aux employés constituant l'équipe projet.

L'onglet « Profil des études » permet de renseigner le profil d'un groupe d'étude, *i.e.* les jours calendaires pour lesquels des actes médicaux sont réalisés (*cf.* Figure 8.12). Le positionnement calendaire du profil varie d'un groupe à l'autre, mais la structure du profil peut également être légèrement différente d'un groupe à l'autre, ce qui justifie le fait que le profil de l'étude soit renseigné au niveau du groupe et non au niveau de l'étude.

L'onglet « Tâches des études » et l'un des onglets les plus importants de l'application car il per-

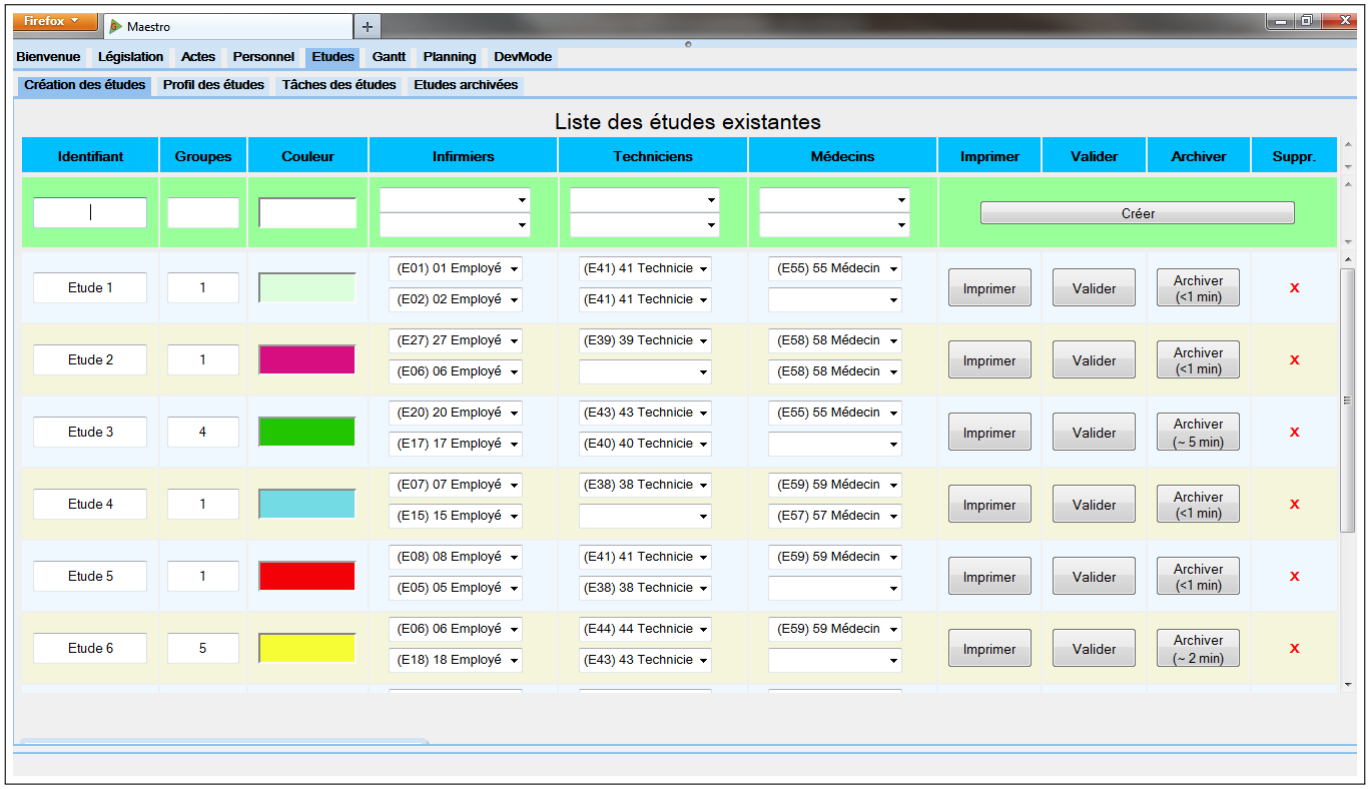


FIGURE 8.11 – Capture d'écran de l'onglet « Création des études ».

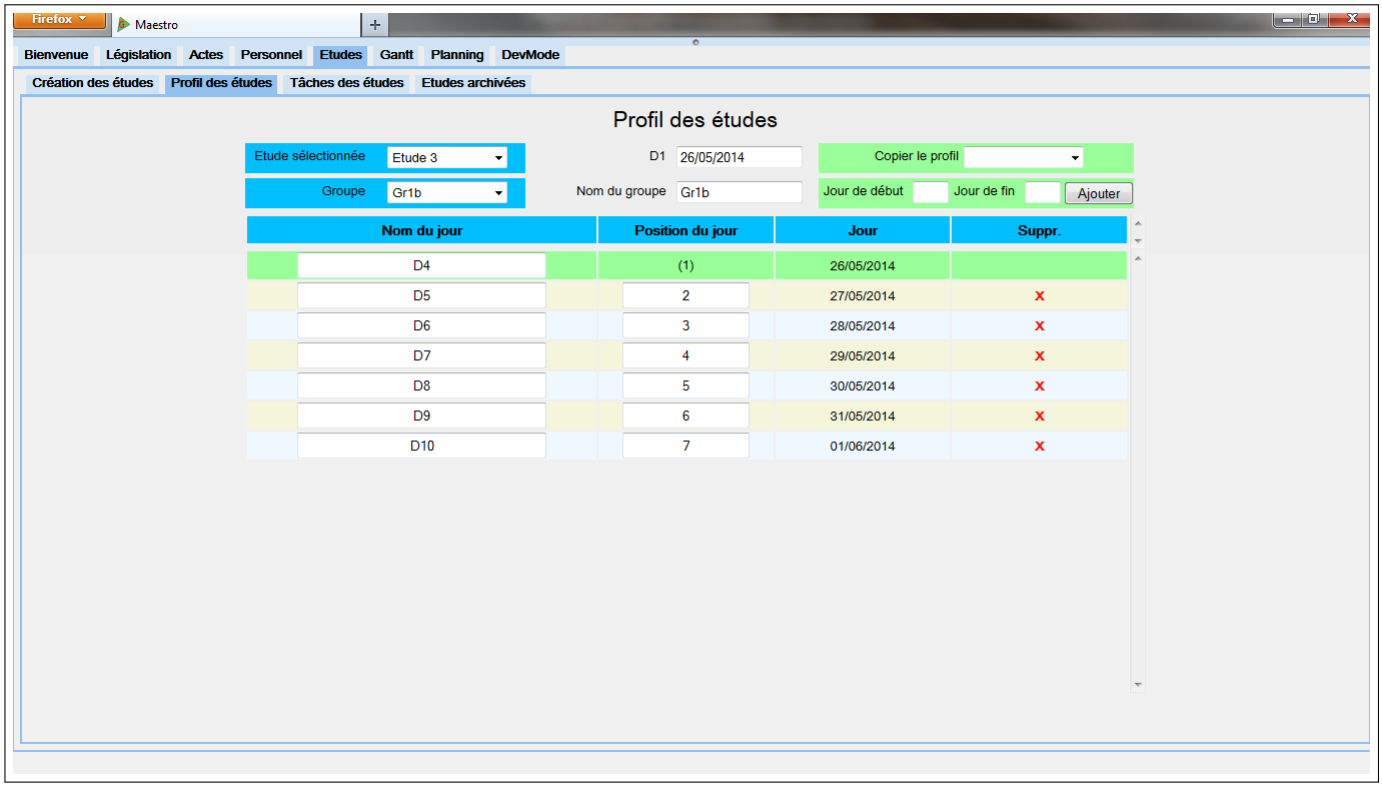


FIGURE 8.12 – Capture d'écran de l'onglet « Profil des études ».

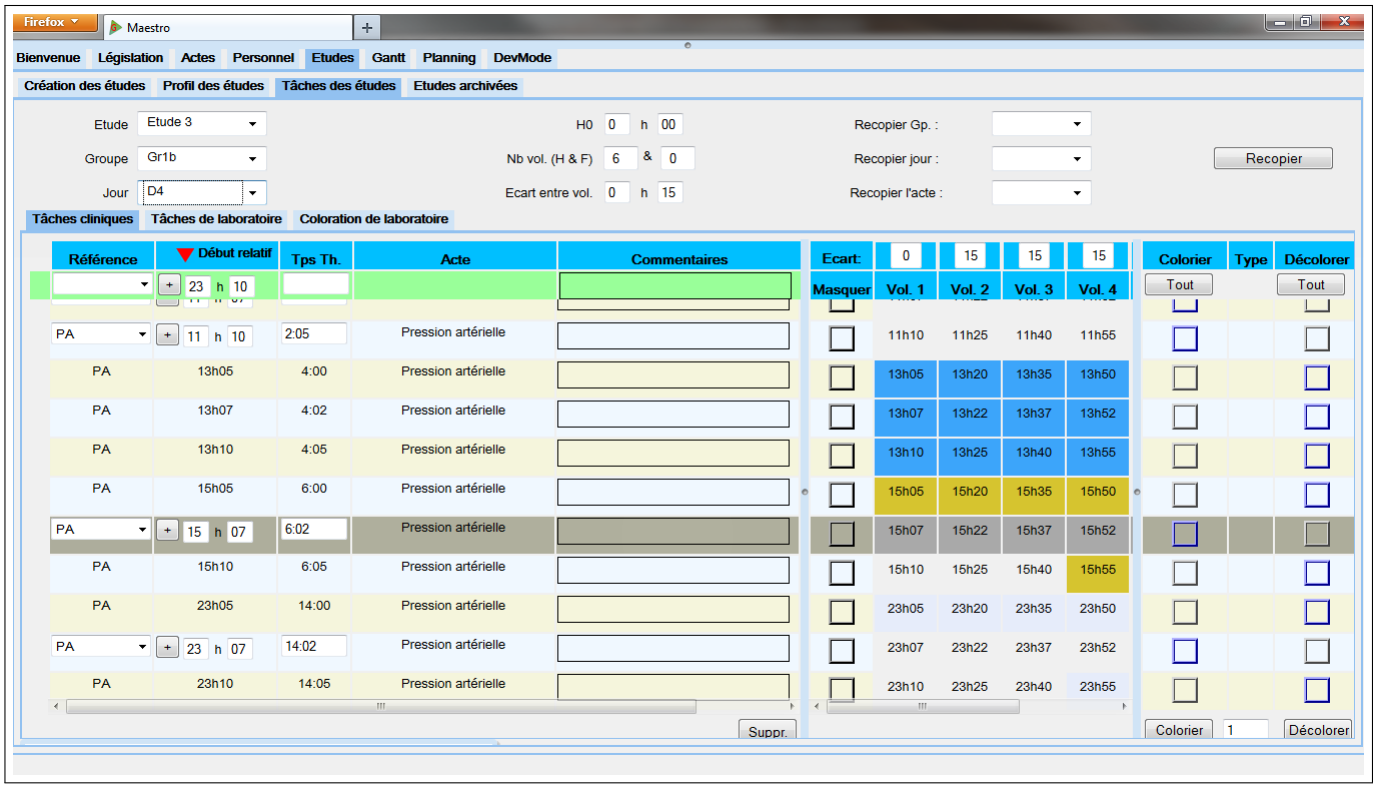


FIGURE 8.13 – Capture d'écran de l'onglet « Tâches des études ».

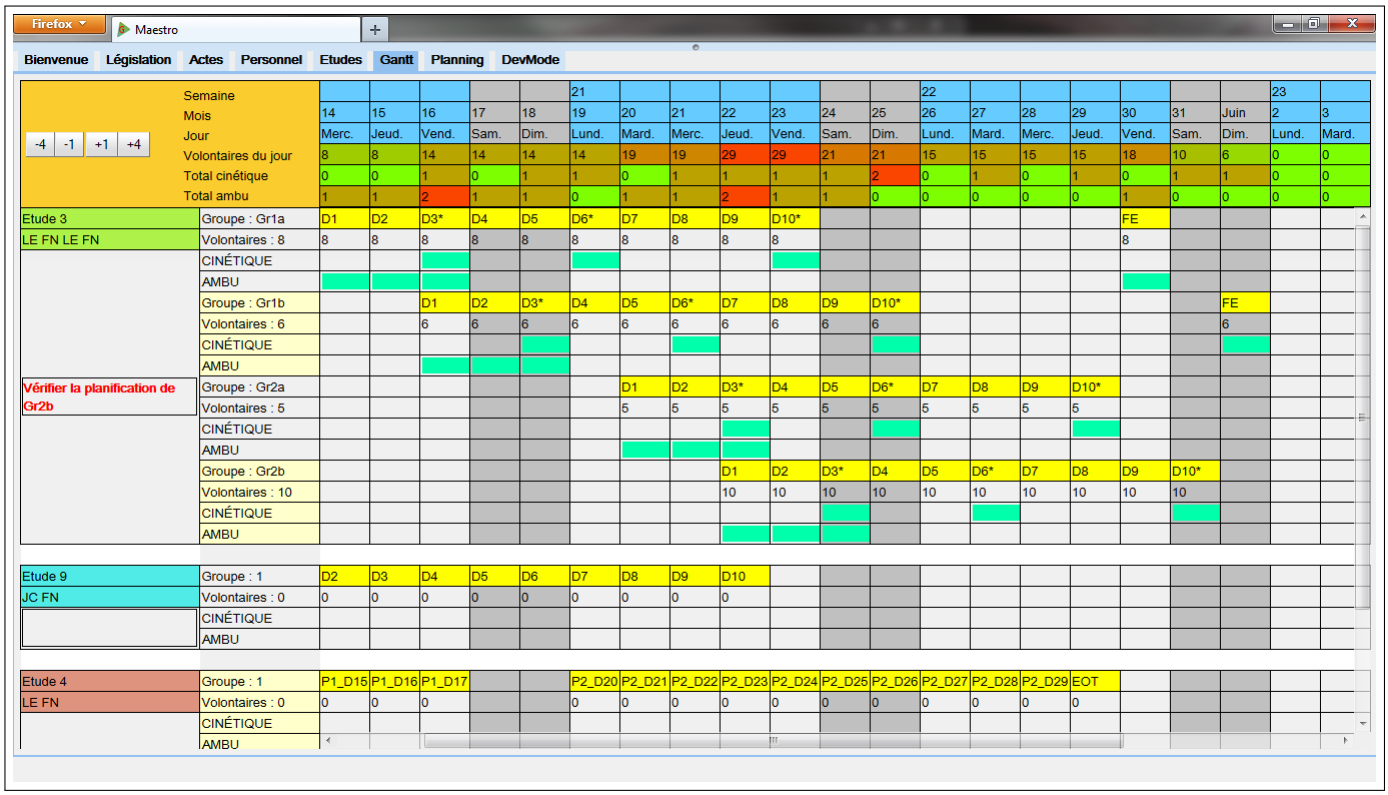


FIGURE 8.14 – Capture d'écran de l'onglet « Gantt ».

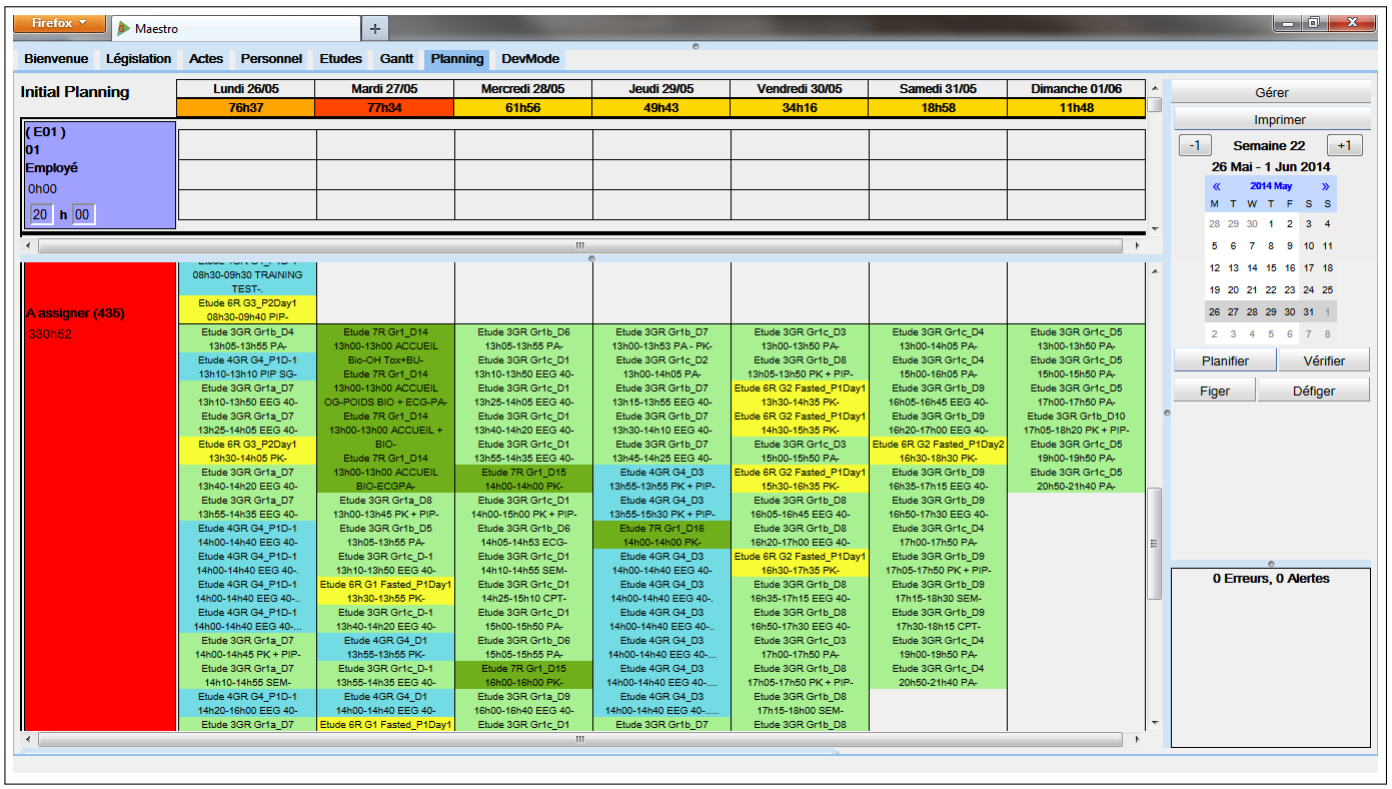


FIGURE 8.15 – Capture d'écran de l'onglet « Planning », avant planification.

met de renseigner les actes médicaux qui se déroulent pour chaque jour de chaque groupe-étude (*cf.* Figure 8.13). Cet onglet permet également de regrouper les actes médicaux en tâches médicales au moyen d'un système de coloration. Cet onglet est particulièrement critique car il a vocation à être utilisé intensivement par tous les employés de Biotrial. Pour cette raison, une attention particulière a été accordée à l'ergonomie de cette interface.

L'onglet « Gantt » permet de visualiser les groupes-études en cours sur un horizon donné (*cf.* Figure 8.14), ce qui permet tout d'abord de vérifier que tous les jours d'études sont correctement positionnés. Par ailleurs, un système de coloration graduelle permet de repérer facilement les jours les plus chargés, à la fois vis-à-vis du nombre de volontaires présents, mais également par rapport à l'intensité des journées de travail en identifiant les jours de cinétique (*cf.* Section 8.1.2, p. 124). Par conséquent, cet onglet contribue également à simplifier le dimensionnement d'équipe en visualisant la charge prévue ou, lorsque les décideurs en ont la possibilité, en lissant l'activité via un changement des dates de début des études.

L'onglet « Planning » permet de gérer toutes les interactions avec le planning du personnel. Tout d'abord, il est possible de visualiser la trame du planning (*i.e.* les congés et les réunions) ainsi que les actes médicaux à réaliser. De plus, un système de coloration graduelle permet d'évaluer la charge de travail globale (*cf.* Figure 8.15). Il est bien entendu possible d'utiliser le moteur de planification automatique de manière à affecter rapidement un maximum de tâches (*cf.* Figure 8.16). Une fois qu'une première solution est calculée, il est possible de la modifier en déplaçant les tâches d'un employé à un autre. Une tâche déplacée à la main est automatiquement figée par Maestro, ce qui signifie qu'une nouvelle planification ne déplacerait pas cette tâche. Bien entendu, il est possible de « défiger » n'importe quelle tâche, *i.e.* retirer la contrainte de pré-affectation portant sur cette tâche. L'idée sous-jacente est qu'un déplacement manuel se fonde sur une expertise métier qui échappe à l'application. Autrement dit, il est préférable de prendre en compte cette pré-affectation pour chaque planification. Pour des raisons d'ergonomie, une tâche figée perd alors sa coloration initiale de ma-

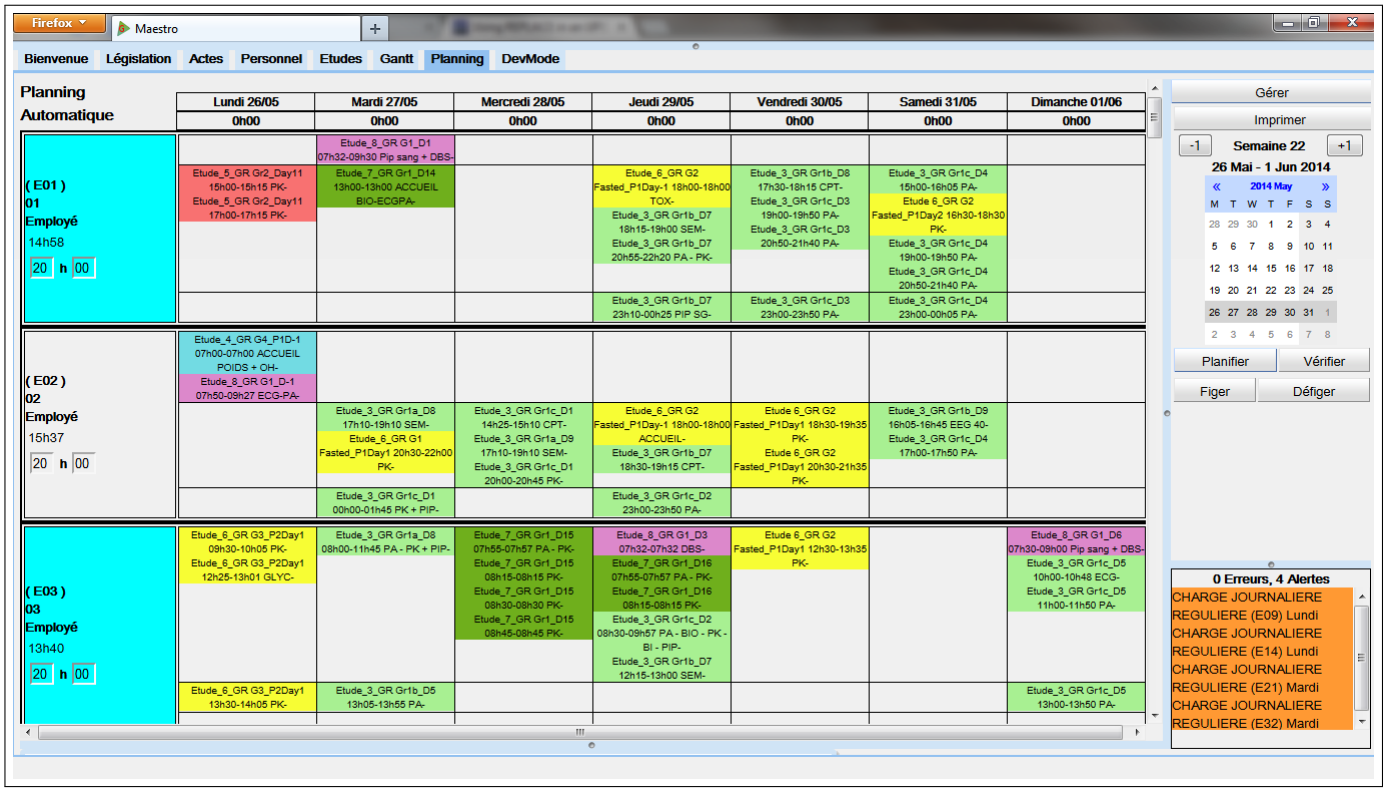


FIGURE 8.16 – Capture d'écran de l'onglet « Planning », après planification.

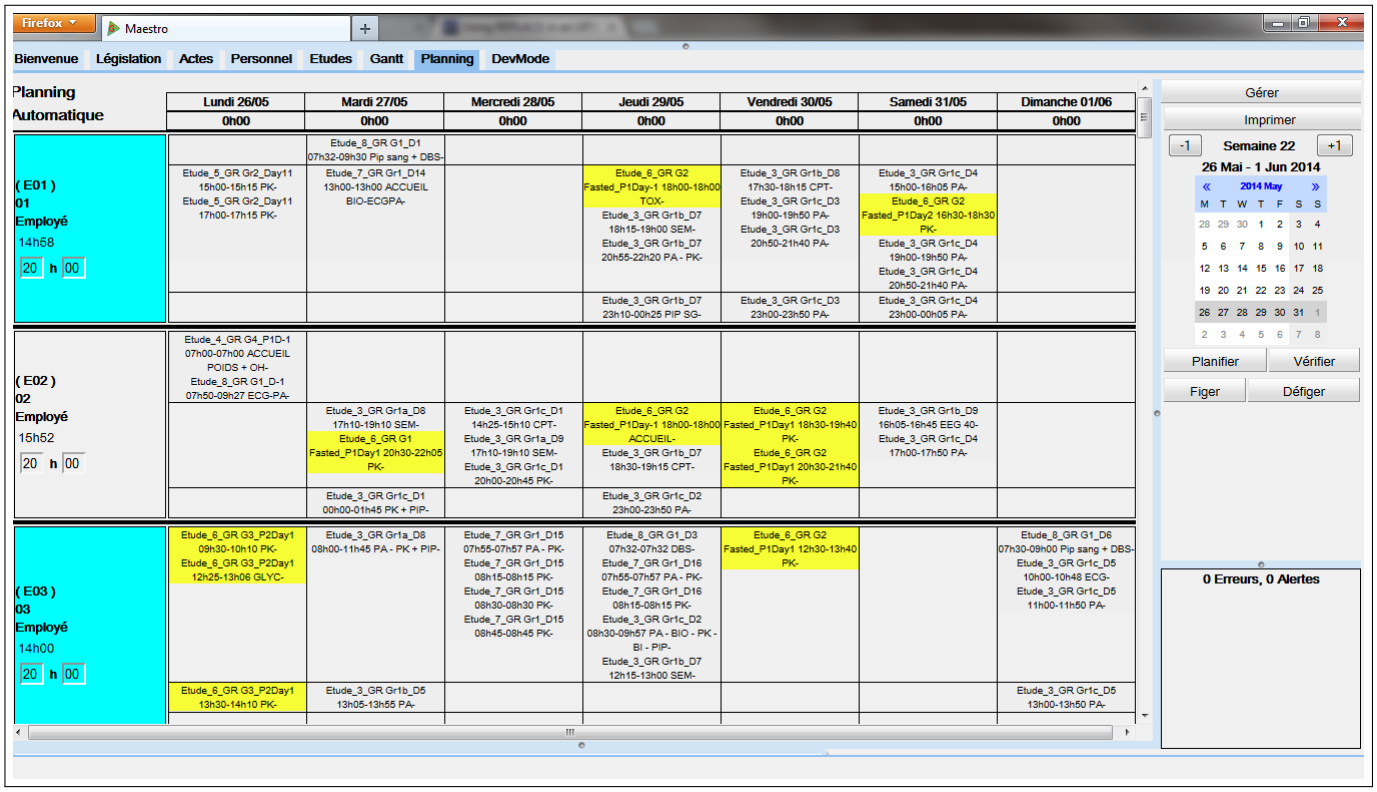


FIGURE 8.17 – Capture d'écran de l'onglet « Planning », mise à jour d'un planning.

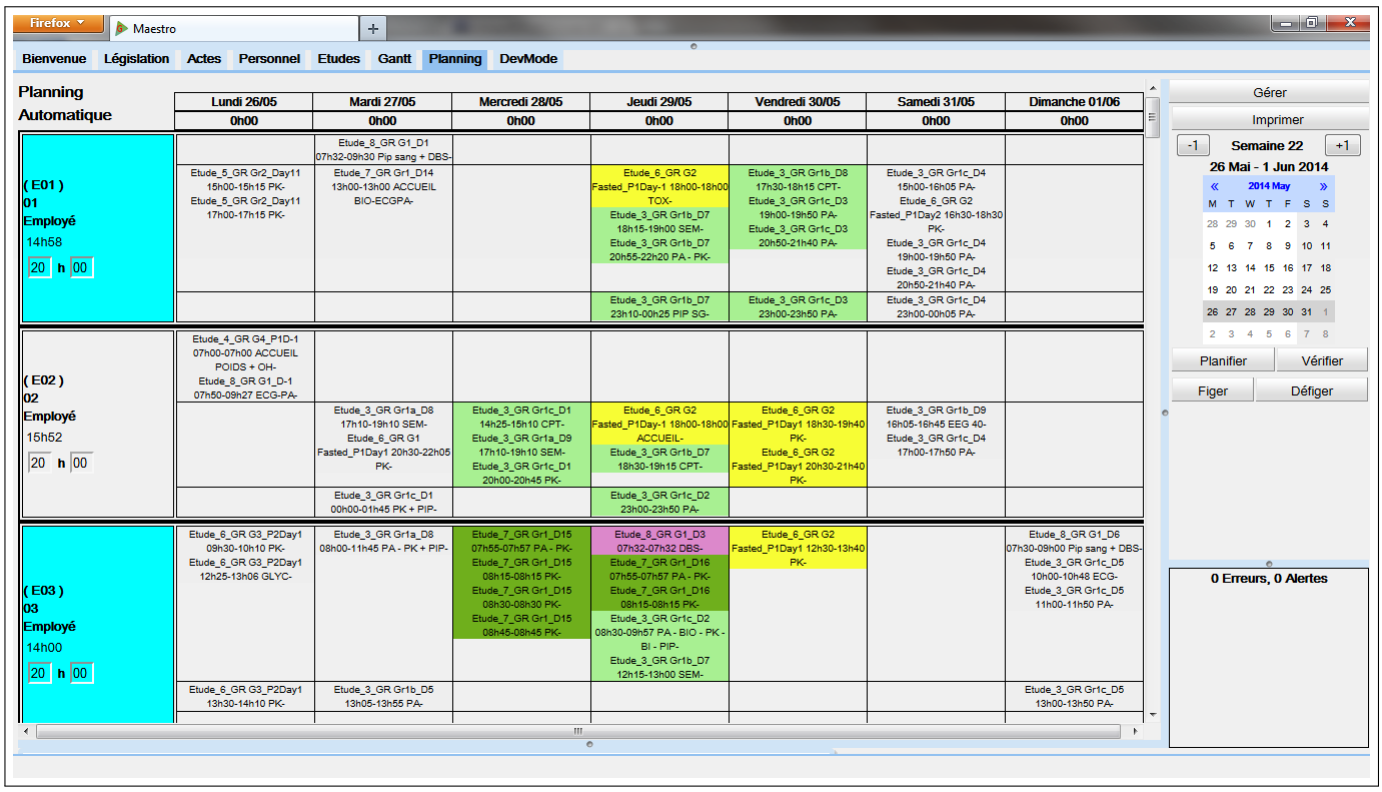


FIGURE 8.18 – Capture d'écran de l'onglet « Planning », planification d'un sous-ensemble de tâches.

nière à signifier que cette tâche ne sera plus déplacée. Par ailleurs, un système de notification sous la forme d'alertes et d'erreurs est également disponible et permet de vérifier la validité et la qualité du planning. Ce mécanisme peut être utilisé à tout moment par le décideur, notamment suite à une série de modifications manuelles. Afin de juger rapidement de la répartition du travail, un code couleur permet d'identifier aisément les employés qui sont au-dessus de leur temps cible, ou au contraire en dessous.

Comme nous avons pu le voir à la section précédente, il est essentiel de mettre à jour les plannings du personnel en fonction des modifications apportées aux données d'entrée, comme par exemple le déplacement d'un ensemble de tâches. Dans ce cas, les infirmières en chef souhaitent que l'ancien planning soit mis à jour, sans pour autant que l'application modifie l'affectation précédente. Cela peut entraîner des problèmes de réalisabilité, et il est par conséquent important de faire ressortir clairement les modifications apportées, pour que les infirmières en chef puissent les gérer de manière adéquate. Afin de faciliter cette procédure de vérification et, si besoin, la replanification, l'application fige automatiquement toutes les tâches qui n'ont pas été modifiées (*cf.* Figure 8.17). Les tâches modifiées apparaissent ainsi clairement. De plus, en cas de replanification, seules les tâches modifiées seront déplacées. Remarquons que les options permettant de figer et de défiger les tâches peuvent être appliquées directement au niveau d'une journée complète, ou au niveau d'un employé. De cette manière, les infirmières en chef peuvent aisément replanifier certains jours sans modifier le reste de la semaine, ou au contraire replanifier certains employés sur toute la semaine, mais sans modifier le planning des autres employés. Par exemple, nous pouvons voir à la Figure 8.18 que les tâches du mercredi, du jeudi et du vendredi peuvent être déplacées par le moteur, alors que les autres tâches sont toutes figées.

8.2.3 Moteur de planification

Le cœur du moteur de planification reprend l'idée générale de la méthode de recherche par voisinage large présentée au Chapitre 7, p. 107. Quelques modifications ont cependant été apportées aux contraintes de la Section 4.1.2 ainsi qu'à la descente locale H_{Δ} (*cf.* Section 4.2.3, p. 66) de manière à prendre en compte certains éléments métiers qui ont émergé au fil des phases de tests de MAESTRO. Nous avons ajouté un temps de déplacement et de préparation entre chaque tâche de 10 minutes. Cela permet aux employés de se déplacer d'un bâtiment à un autre, ce qui est devenu important puisque Biotrial est actuellement en train d'étendre ses locaux. Sans cette contrainte de déplacement, le moteur de planification attribuait fréquemment des tâches se suivant de très près aux employés, ce qui ne convenait pas. Par ailleurs, la contrainte sur la pause déjeuner a également été renforcée de manière à garantir une pause déjeuner non morcelée. Les infirmières en chef préfèrent en effet que le moteur respecte cette contrainte, même si durant la phase de modification manuelle, les infirmières en chef peuvent éventuellement réduire ce temps de pause si besoin. Cette modification de la contrainte C 14 (*cf.* 4.1.2, p. 60) revient en réalité à remettre en question la restriction R 5 (*cf.* 2.2.4, p. 32). Par conséquent, nous avons cherché à évaluer a posteriori la pertinence de cette restriction en calculant le pourcentage de pauses déjeuner non morcelées par rapport au nombre total de pauses déjeuner requises. L'analyse des meilleurs résultats obtenus avec la LNS, nous a ainsi permis de constater que 97,10% des vacations éligibles à la pause déjeuner présentait une pause déjeuner entière durant la vacation. Dans la grande majorité des cas (85,7%), ces pauses sont de plus positionnées au niveau de la fenêtre horaire du déjeuner. Autrement dit, en moyenne, l'hypothèse de départ semble cohérente par rapport aux résultats obtenus. Néanmoins, cette moyenne peut cacher des cas pathologiques qui ne sont pas appréciés par les employés. Par ailleurs, le contexte actuel de modification des pratiques de travail et de rapatriement de l'activité parisienne sur le site de Rennes est, en soit, générateur d'inconforts pour les employés, ce qui nous a conduit à revoir l'hypothèse initiale selon laquelle les pauses déjeuner pouvaient être morcelées. Pour finir, en plus de la répartition équitable de la charge de travail, nous avons ajouté des critères additionnels à la fonction objectif :

- Le nombre de personnes réalisant des horaires de nuit doit être le plus faible possible.
- Les horaires de travail dépassant le temps de travail habituel sont à éviter.
- Les infirmiers doivent réaliser en priorité les tâches cliniques.

Il est difficile de pondérer l'importance de ces différents critères les uns par rapport aux autres, car cela dépend de nombreux éléments métier, tels que les activités médicales à réaliser et la charge de travail global. Actuellement, ces différents éléments sont agrégés au sein d'une somme pondérée, mais le paramétrage de ces différents coefficients requiert encore quelques itérations avec Biotrial.

8.3 Retour sur le projet MAESTRO

Nous présentons ici les grandes étapes du projet MAESTRO, récapitulées aux Figures 8.19 et 8.20. Nous dressons ensuite un bilan du projet en nous intéressant plus particulièrement au logiciel développé pour Biotrial. Suite à ce bilan, nous discutons des perspectives éventuelles de ce projet, tant sur le plan académique que du point de vue de Biotrial.

8.3.1 Déroulement du projet

Le projet MAESTRO a mobilisé un nombre important de personnes sur une durée de quatre ans, à la fois du côté de Biotrial et du côté de l'équipe de recherche et de développement. Du côté de Biotrial, plusieurs réflexions sur la standardisation des données et des procédures ont été menées. De plus, des phases de tests régulières ont été réalisées afin de valider l'ergonomie et les fonctionnalités de MAESTRO. Par ailleurs, le service informatique de Biotrial s'est occupé d'installer, configurer

et maintenir les serveurs de l'application. En effet, pour des raisons de sécurité, Biotrial a choisi un hébergement en interne. De son côté, l'équipe de développement s'est chargée de l'étude de faisabilité, de la compréhension du besoin et de la rédaction des spécifications techniques du logiciel. L'équipe a également pris en charge une grande partie du développement et du suivi de projet. Au fil de l'avancée des travaux de recherche, nous avons mis à jour le moteur de résolution utilisé par l'application, et nous l'avons adapté au fur et à mesure de nos échanges avec Biotrial. Les nombreux sprints de développement ont été réalisés par l'équipe de développement, en collaboration avec un développeur professionnel et un manager d'équipe (Sprints 0 à 5), ainsi qu'avec deux stagiaires (Sprints 11 à 13).

Les différentes étapes du projet MAESTRO apparaissent sur le diagramme de Gantt (*cf.* Figures 8.19 et 8.20). Notons que les activités de recherche réalisées durant la thèse n'apparaissent pas sur ces diagrammes. Nous pouvons constater qu'une part importante des développements fut réalisée dans la seconde moitié de l'année 2012. Ces développements nous ont permis de mettre en place la structure générale de l'application, ainsi que les fonctionnalités essentielles. Plus précisément, nous avons créé les interfaces de saisie des données d'entrée pour les contrats, les actes, les employés, les études et les tâches cliniques, ainsi que l'interface de visualisation et d'interaction avec les plannings. À ce stade, une première version du moteur était déjà fonctionnelle, mais l'essentiel des retours utilisateurs portaient sur l'ergonomie et les fonctionnalités de l'interface de saisie des données. Durant l'année 2013, nous avons amélioré et complété l'interface de saisie des données, notamment en repensant la gestion des tâches techniques. Nous avons ensuite déployé la nouvelle version du moteur sur MAESTRO et nous avons réalisé plusieurs tests pour évaluer la qualité des plannings. Au terme de cette phase de test, nous avons décidé de renforcer la contrainte sur la pause déjeuner et d'ajouter la contrainte sur le temps de déplacement minimum entre deux tâches. Par ailleurs, nous nous sommes également rendu compte que plusieurs fonctionnalités importantes devaient être ajoutées à l'interface de saisie, de manière à gérer certains cas qui avaient été écartés en début de projet, mais qui se sont révélés très importants par la suite.

Au début de l'année 2014, nous avons fait un point intermédiaire avec notre partenaire. À cette occasion, nous avons identifié plusieurs éléments majeurs présentant une marge de progression importante. En premier lieu, l'application présentait des temps de chargement trop longs. De plus, l'interface de saisie des données ne permettait pas de gérer certains cas réels apparaissant de plus en plus souvent. Pour finir, l'application était également trop contraignante vis-à-vis de la mise à jour obligatoire des plannings, qui avait été mis en place de manière à garantir l'intégrité et la cohérence de toutes les données. Pour remédier à cela, nous avons tout d'abord accéléré le chargement des données, nous avons ensuite modifié les bases de données et l'interface pour gérer tous les cas particuliers répertoriés à ce jour, et nous avons également repensé les fonctionnalités de mise à jour des plannings du personnel. Ces évolutions ont permis à l'application d'atteindre un état de stabilité et d'efficacité suffisant pour envisager une mise en production à moyen terme. La fin du projet MAESTRO permettra de mettre en place un transfert de compétence entre l'équipe qui a développé MAESTRO et le service informatique de Biotrial qui assurera sa maintenance.

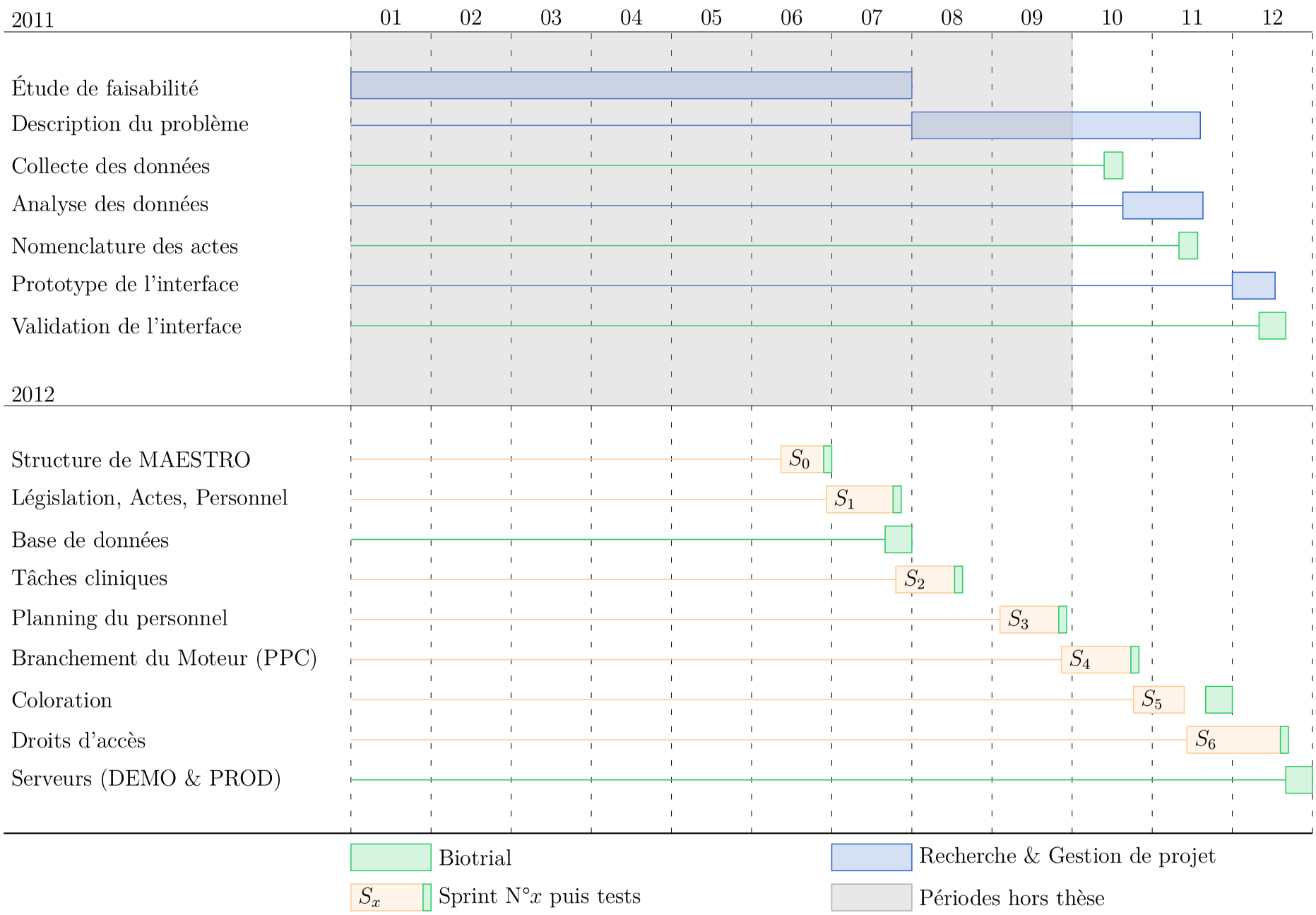


FIGURE 8.19 – Gantt du projet MAESTRO, années 2011 et 2012

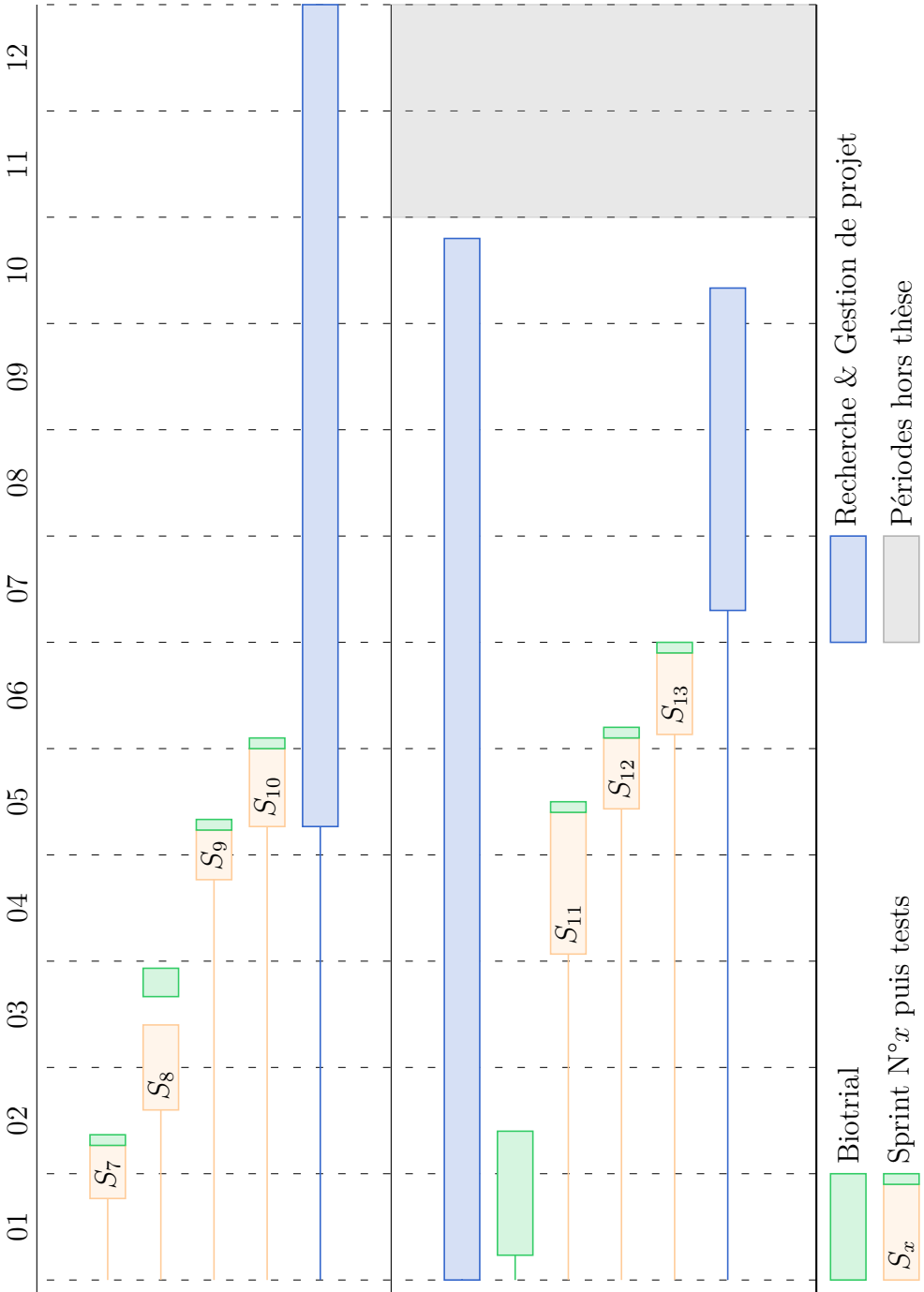


FIGURE 8.20 – Gantt du projet MAESTRO, années 2013 et 2014

8.3.2 Validation itératives des fonctionnalités

De manière à maximiser les chances de succès du projet, nous avons opté pour un mode de développement à la fois incrémental et itératif, inspiré des méthodes agiles. Ces méthodes partent du principe que tout projet comporte une part d'incertitude impossible à prédire complètement. Par exemple, le demandeur peut changer d'avis au fil du projet, soit parce que certains éléments avaient échappé à la phase d'analyse et de spécification, soit car le simple fait de construire une première solution permet de se projeter plus efficacement dans la suite du projet et ainsi entrevoir de nouvelles contraintes ou opportunités. Par ailleurs, le contexte socio-économique et législatif peut également évoluer indépendamment de la volonté du demandeur. Pour finir, certaines difficultés techniques peuvent avoir été sous-estimées et des incompréhensions initiales peuvent conduire le projet dans une impasse. Pour toutes ces raisons, les méthodes agiles ne cherchent pas à prédire complètement l'avenir mais à le contrôler. En effet, plus l'équipe en charge du projet est en capacité de le réorienter au fil du temps, moins il est important de prévoir précisément chaque étape dès le début du projet. La formule suivante résume bien la philosophie de ce mode de développement :

« *Mieux vaut être grossièrement juste que précisément faux.* » Alistair Cockburn

Autrement dit, le problème n'est plus de prévoir parfaitement tout le déroulement du projet de manière à atteindre le but initial, mais plutôt de redéfinir régulièrement les objectifs à court terme de manière à faire converger le projet vers une issue favorable, potentiellement différente des objectifs initiaux.

Durant le projet MAESTRO, nous avons réalisé une quinzaine de réunions avec les futurs utilisateurs de MAESTRO, ainsi que les services informatique et qualité. Ces réunions nous ont permis de valider les fonctionnalités développées, de les corriger et de les modifier. Cela nous a également permis de revoir constamment le périmètre de l'application ainsi que les différentes priorités. Par exemple, nous avons rapidement conclu que la gestion des services pharmacie et accueil, beaucoup moins stratégiques et contraints que les services techniques et cliniques, n'était pas prioritaire. Par ailleurs, certaines hypothèses établies au début du projet ont finalement dû être supprimées en raison de l'évolution des pratiques métier de Biotrial. Cela nous a conduit à réaliser des changements importants mais essentiels, à la fois sur la base de données, l'interface graphique mais également sur les services et fonctionnalités de l'application.

8.3.3 Réflexion sur les process

Le développement de MAESTRO a permis de faire le point sur les process de Biotrial, récemment chamboulés par la réorganisation de l'entreprise et l'augmentation de l'activité. À cette occasion, il est devenu évident que certains process gagneraient à être adaptés pour mieux correspondre aux nouveaux besoins de Biotrial. Il est par exemple de plus en plus fréquent d'anticiper sur les autorisations de lancement des tests pharmaceutiques. Cette pratique, autrefois exceptionnelle, tend à se généraliser, ce qui permet à Biotrial d'obtenir des délais plus concurrentiels face aux offres concurrentes qui ne sont pas soumises aux mêmes contraintes administratives. Cela conduit néanmoins à une incertitude sur la date de démarrage des études et sur l'activité des employés. Cette incertitude est actuellement palliée via un travail d'anticipation de la part des infirmières planificatrices qui prennent alors en charge une part du travail de l'équipe-projet. Une rationalisation de ce process permettrait sans doute de gagner en efficacité et apporterait plus de sérénité aux équipes de travail.

La mise en place de MAESTRO peut demander un certain temps d'adaptation de la part des décideurs et du personnel, mais cela devrait permettre, à terme, de gagner du temps et de la réactivité. En effet, il faut actuellement plusieurs jours pour centraliser toutes les informations nécessaires à la planification d'une semaine de travail et pour répartir les tâches médicales sur les employés de manière

à obtenir une première version du planning. MAESTRO peut générer un planning automatique en quelques minutes, laissant ainsi du temps pour ajuster ce planning en réaffectant les quelques tâches difficiles à positionner, ou dont l'affectation automatique ne convient pas tout à fait au regard des quelques pratiques métier non prises en compte par le logiciel. De la même manière, lorsqu'un imprévu survient, il est fastidieux de décaler ou réaffecter toutes les tâches à la main tout en vérifiant à nouveau la validité de chaque planning. Par rapport à cette difficulté, MAESTRO facilite la mise à jour des plannings en modifiant automatiquement toutes les tâches et en soulignant les modifications réalisées, permettant ainsi au décideur une vérification ciblée des contraintes métier. Toutes les contraintes prises en compte par MAESTRO peuvent être vérifiées automatiquement suite à une mise à jour du planning ou une modification manuelle. Par ailleurs, MAESTRO permet de tester rapidement plusieurs scénarios vis-à-vis des embauches d'intérimaires. Il est par exemple possible de compléter l'effectif de travail avec plusieurs intérimaires et de faire varier leur charge médicale idéale ainsi que leur horizon d'embauche pour converger vers une solution requérant un minimum d'intérimaires.

8.3.4 Mise en place de MAESTRO

Les futurs utilisateurs de MAESTRO sont satisfaits de la version actuelle du logiciel : le périmètre fonctionnel de l'application permet de répondre correctement au besoin initial et le logiciel est suffisamment rapide et agréable pour être utilisé au quotidien.

Puisque MAESTRO a été conçu comme un système autonome et indépendant, sa mise en place ne risque pas de perturber le système actuel, ce qui signifie qu'il sera possible d'utiliser les deux systèmes durant toute la phase d'intégration. Cette intégration en doublon présuppose une forte disponibilité des infirmières en chef. Pour cela, Biotrial envisage une intégration par étape, en commençant par le service ambulatoire qui est actuellement rattaché au service clinique, mais dont l'activité a vocation à être gérée à part entière. L'antenne de New York pourrait également offrir un environnement de test propice à l'application, puisque le volume d'études réalisées par cette antenne est bien plus faible qu'à Rennes. De cette manière, cela permettrait de corriger les derniers problèmes restants dans un contexte plus souple, de manière à envisager sereinement un déploiement plus large à moyen terme.

Pour finir, en plus des nombreuses phases de tests réalisées tout au long du projet, nous avons prévu plusieurs jours de formation sur MAESTRO, à la fois vis-à-vis du service informatique de Biotrial qui assurera par la suite la maintenance de l'application, mais également vis-à-vis des utilisateurs.

8.3.5 Perspectives

Comme nous l'avons signalé à la Section 2.3.1, p. 32, Biotrial ne disposait pas, en début de projet, des données nous permettant de tester le moteur de planification en situation réelle. Au fur et à mesure du développement de l'application, nous avons collecté des jeux de données, mais nous n'avons pas pu les utiliser comme base de comparaison en raison d'un biais sur les données d'entrée. Nous nous sommes rendu compte à cette occasion que la structure de la base de données devait être adaptée de manière à ce que les données saisies sur MAESTRO correspondent bien aux données réelles de Biotrial, alors que cela n'était pas possible auparavant. Cette modification importante a été réalisée avec succès, mais trop tardivement pour que nous puissions collecter de nouvelles données. Par conséquent, il serait très utile et intéressant de collecter de nouvelles données de manière à évaluer les gains de temps lors de la planification du personnel ainsi que les gains financiers relatifs aux embauches d'intérimaires. Il serait également envisageable de comparer les plannings automatiques avant et après modification manuelle de manière à améliorer le moteur.

9

Résolution du problème tactique

Dans ce chapitre, nous nous éloignons du problème opérationnel de Biotrial, au profit de la problématique tactique, que nous abordons au travers du *Shift Minimisation Personnel Task Scheduling Problem* (SMPTSP). Nous rappelons tout d'abord la définition du SMPTSP et montrons l'intérêt de ce problème dans le cadre de notre étude. Par la suite, nous proposons une modélisation PPC, puis nous montrons qu'il est possible de renforcer ce modèle en tirant parti de la structure du problème. Nous présentons une stratégie de branchement et nous évaluons les performances de l'approche sur plusieurs jeux d'instances. Ces travaux ont été réalisé en collaboration avec Jean-Guillaume Fages, doctorant de l'équipe TASC (***T**héorie, **A**lgorithmes et **S**ystèmes en **C**ontraintes*, Laboratoire d'informatique de Nantes Atlantique, École des Mines de Nantes).

Sommaire

9.1 Description du SMPTSP

9.1.1 Notations et modélisation PLNE

Soient $\mathcal{T}$ et $\mathcal{S}$ deux ensembles correspondant respectivement à un ensemble de tâches et de ressources. Chaque tâche $t \in \mathcal{T}$ est associée à un ensemble de ressources affectables, noté $\mathcal{S}_t \subseteq \mathcal{S}$. Le SMPTSP consiste à affecter à chaque tâche $t \in \mathcal{T}$, une ressource $s \in \mathcal{S}_t$, de manière à minimiser le nombre total de ressources mobilisées, sachant qu'une même ressource ne peut être affectée à deux tâches concomitantes et qu'une ressource affectée à une tâche doit réaliser toute la tâche. Autrement dit, le SMPTSP correspond à un problème de coloration de listes dans un graphe d'intervalles, et il s'agit donc d'un problème NP-Difficile [92]. En revanche, dans le cas particulier où chaque ressource peut réaliser toutes les tâches, *i.e.* $\mathcal{S}_t = \mathcal{S}, \forall t \in \mathcal{T}$, le SMPTSP revient alors à un problème de coloration de sommets dans un graphe d'intervalles, et peut donc se résoudre en temps polynomial [92].

Puisque les tâches concomitantes doivent nécessairement être affectées à des ressources différentes, le nombre maximal de tâches concomitantes, noté $\mathbf{M//}$, donne une borne inférieure sur le nombre d'employés requis pour réaliser toutes les tâches de $\mathcal{T}$. Dans un graphe d'intervalles, l'ensemble des cliques maximales, noté $\mathcal{C}$, peut être obtenu en temps polynomial [101]. Ces cliques maximales permettent d'exprimer les contraintes de non ubiquité des ressources de manière synthétique. Par ailleurs, la taille de la plus grande clique correspond à M//.

M// : nombre maximal de tâches concomitantes parmi l'ensemble des tâches de $\mathcal{T}$. Dans le graphe d'intervalles correspondant, ce nombre correspond à la cardinalité d'une clique maximum.

Nous rappelons à présent la modélisation PLNE proposée par Krishnamoorthy *et al.* pour le SMPTSP [127], que nous notons *MIP model*. Cette modélisation repose sur les variables binaires $x_{t,s}$ et y_s, qui indiquent respectivement si la tâche t est assignée à la ressource s et si la ressource s est utilisée. Les contraintes (9.2) garantissent que toutes les tâches sont affectées à des ressources qui leur sont affectables. Les contraintes (9.3) vérifient tout d'abord qu'aucune ressource n'est affectée à deux tâches concomitantes, mais également que chaque ressource affectée à au moins une tâche est bien considérée comme étant utilisée. À partir de là, l'objectif du problème est pris en compte via (9.1).

9.1.2 Exemple de référence

Nous proposons une illustration du SMPTSP à la Figure 9.1. Cet exemple sert de fil conducteur tout au long de ce chapitre. Les données d'entrée sont représentées à la Figure 9.1a : il s'agit ici d'un ensemble de 5 tâches $\mathcal{T} = \{t_1, .., t_5\}$ et d'un ensemble de 5 ressources $\mathcal{S} = \{s_1, .., s_5\}$. Chaque tâche, représentée par un rectangle, est associée à un ensemble de ressources affectables, *e.g.* la tâche t_1 est affectable aux ressources $\{s_2, s_3, s_4\}$ ($\mathcal{S}_{t_1} = \{s_2, s_3, s_4\}$). Les dates de début et de fin des tâches

MIP model :

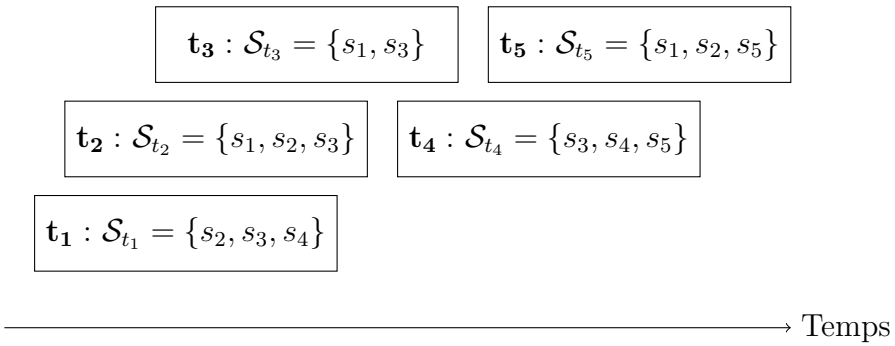
$$\min \quad \sum_{s \in \mathcal{S}} y_s \tag{9.1}$$

$$s.t. \quad \forall t \in \mathcal{T}, \quad \sum_{s \in \mathcal{S}_t} x_{t,s} = 1 \tag{9.2}$$

$$\forall s \in \mathcal{S}, \forall \mathcal{K} \in \mathcal{C}, \quad \sum_{t \in \mathcal{K}} x_{t,s} \leq y_s \tag{9.3}$$

$$\forall t \in \mathcal{T}, \forall s \in \mathcal{S}_t, \quad x_{t,s} \in \{0, 1\} \tag{9.4}$$

$$\forall s \in \mathcal{S}, \quad y_s \in \{0, 1\} \tag{9.5}$$



(a) Données d'entrée.

Tâches	Ressources
t_1	s_2
t_2	s_3
t_3	s_1
t_4	s_3
t_5	s_1

(b) Solution optimale avec 3 ressources.

FIGURE 9.1 – Exemple d'un SMPTSP avec 5 tâches et 5 ressources

permettent d'obtenir l'ensemble des cliques maximales $\mathcal{C} = \{\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3\}$ avec $\mathcal{K}_1 = \{t_1, t_2, t_3\}$, $\mathcal{K}_2 = \{t_1, t_3, t_4\}$, $\mathcal{K}_3 = \{t_4, t_5\}$. Par conséquent, M// = 3. Une solution optimale impliquant les ressources $\{s_1, s_2, s_3\}$ est donnée à la Figure 9.1b.

9.1.3 Liens avec le SDPTSP||U,Δ

Le SMPTSP diffère du SDPTSP||U,Δ sur deux points majeurs. Premièrement, les contraintes légales ne sont pas prises en compte. Cela revient en réalité à considérer que les horaires de travail sont fixés en amont du SMPTSP, soit par un décideur, soit par une méthode de construction des horaires du personnel. Deuxièmement, l'objectif n'est pas de trouver une solution complète et équitable mais une solution qui minimise le nombre d'employés permettant de couvrir toutes les tâches. Notons par ailleurs que le nombre maximum d'employés disponibles est toujours suffisant pour couvrir toutes les tâches.

Dans le cadre de notre étude, le SMPTSP est intéressant sur un plan tactique essentiellement. Il serait envisageable, par exemple, de concevoir une méthode itérative en deux phases reposant sur une résolution efficace du SMPTSP de manière à obtenir une borne inférieure sur le nombre d'intérimaires. Une première étape consisterait à compléter l'effectif permanent de Biotrial via un large ensemble d'intérimaires maîtrisant des compétences communes. À partir de là, une première phase permettrait de fixer les horaires de travail des employés sur plusieurs semaines, alors que la seconde phase consisterait à résoudre le SMPTSP correspondant. En itérant entre les deux phases, il serait alors possible de faire converger les horaires du personnel de manière à minimiser le nombre d'intérimaires embauchés. Pour faire converger les horaires du personnel, il serait envisageable d'utiliser un système de voisinage similaire à celui décrit au Chapitre 5.

Cette méthode permettrait d'une part d'évaluer les besoins en intérimaires sur le long terme, mais cela donnerait également des indications sur les compétences minimales requises pour chaque intérimaire.

9.2 Modélisation PPC avec la contrainte `atMostNValue`

Dans cette section nous proposons tout d'abord une modélisation du SMPTSP sous la forme d'un modèle PPC. Cette modélisation repose sur la contrainte `atMostNValue`, dont nous détaillons le fonctionnement en présentant son propagateur le plus efficace : le propagateur *greedy* [27]. Par la suite, nous introduisons un nouveau propagateur, de la même famille que *greedy*. Ce propagateur permet de filtrer efficacement la contrainte `atMostNValue`$(\mathcal{X}, z)$ lorsque des variables de $\mathcal{X}$ doivent

prendre des valeurs différentes.

9.2.1 Modélisation PPC & Fonctionnement de `atMostNValue`

Le SMPTSP peut être formulé en PPC au moyen d'un ensemble de variables entières $\mathcal{X}$ et d'une variable z correspondant à l'objectif du problème. La variable $x_t \in \mathcal{X}$ donne alors la ressource affectée à la tâche t. Plus formellement, si l'on note $d(x_t)$ le domaine de la variable x_t, alors $d(x_t) = \{j\}$ signifie que la tâche t est affectée à la ressource j et $d(z)$ donne le nombre de ressources affectées à au moins une tâche. Nous obtenons alors un modèle PPC, noté *CP model*, qui repose sur deux contraintes globales : la contrainte `allDifferent` est utilisée pour interdire l'affectation de tâches concomitantes à une même ressource (9.8) ; la contrainte `atMostNValue` est utilisée pour contraindre le nombre de ressources utilisées (9.7). Les domaines initiaux des variables sont définis par les contraintes (9.9) et (9.10).

La contrainte `atMostNValue` appartient à la famille des contraintes du type *Number of Distinct Values*. Elle a été introduite par Pachet et Roy [162], mais le premier algorithme de filtrage est proposé par Beldiceanu [22]. Bessière *et al.* [27] ont par la suite proposé plusieurs algorithmes de filtrage pour la contrainte `atMostNValue`. Suite à leur étude, les auteurs établissent que le propagateur *greedy* offre un bon compromis entre filtrage et temps de calcul. Par conséquent, nous prenons ce propagateur comme point de référence. Avant de décrire plus en détail le fonctionnement du propagateur *greedy*, nous rappelons quelques définitions :

Définition 1. *Le graphe d'intersection d'un ensemble de variables $\mathcal{X}$, noté $G_{\mathcal{I}}(\mathcal{X}) = (V, E_{\mathcal{I}})$, est défini par : 1) un ensemble de sommets V où chaque sommet $i \in V$ est associé à une variable $x_i \in \mathcal{X}$, 2) un ensemble d'arêtes $E_{\mathcal{I}}$ représentant l'intersection des domaines des variables. Plus formellement, pour tout couple de sommet $(i, j) \in V^2$, l'existence d'une arête $(i, j) \in E_{\mathcal{I}}$ signifie que l'intersection des variables correspondantes est non vide : $d(x_i) \cap d(x_j) \neq \emptyset$.*

Définition 2. *Un stable dans un graphe $G = (V, E)$ est un sous-ensemble de sommets non adjacents, $A \subseteq V$.*

Définition 3. *Un stable maximal dans un graphe G est un stable qui n'est inclus dans aucun autre stable.*

Définition 4. *Un stable maximum dans un graphe G est un stable de cardinalité maximale. Le nombre de stabilité d'un graphe G, noté $\alpha(G)$, correspond à la cardinalité d'un stable maximum de G. L'ensemble des stables d'un graphe G, est noté $\mathcal{IS}(G)$.*

En reprenant l'exemple donné à la Figure 9.1, nous obtenons un graphe d'intersection complet, ce qui signifie que toute paire de tâches $(t_1, t_2) \in \mathcal{T}^2$ partage au moins une ressource affectable commune (*cf.* Figure 9.2).

CP model :

$$\min \quad z \tag{9.6}$$

$$s.t. \quad \texttt{atMostNValue}(\mathcal{X}, z) \tag{9.7}$$

$$\forall \mathcal{K} \in \mathcal{C}, \quad \texttt{allDifferent}(\{x_i \mid i \in \mathcal{K}\}) \tag{9.8}$$

$$\forall x_i \in \mathcal{X}, \quad d(x_i) = \{j \in \mathcal{S}_{t_i}\} \tag{9.9}$$

$$d(z) = [\mathrm{M}//, |\mathcal{S}|] \tag{9.10}$$

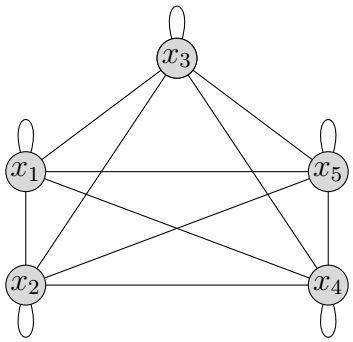


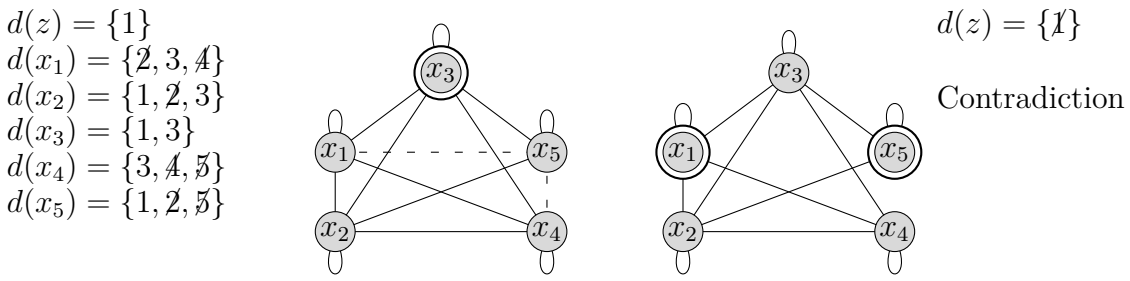
FIGURE 9.2 – Graphe d'intersection, $G_{\mathcal{I}}(\mathcal{X})$, sur notre exemple.

L'algorithme de filtrage proposé par Bessière *et al.* [27] repose sur la recherche d'un stable maximum dans le graphe $G_{\mathcal{I}}(\mathcal{X})$. Puisque ce problème est NP-Difficile [97], les auteurs proposent une heuristique gloutonne permettant de construire un stable maximal A de $G_{\mathcal{I}}(\mathcal{X})$ en itérant entre une phase de sélection d'un sommet de degré minimal et une phase de suppression du sommet sélectionné, de ses voisins et des arêtes correspondantes [105]. Cette heuristique est notée `MD` pour *Minimum Degree*. À partir du stable ainsi obtenu, le propagateur filtre selon les règles $\mathcal{R}_1$ et $\mathcal{R}_2$, avec $\underline{z}$ et $\overline{z}$ qui correspondent respectivement à la borne inférieure et à la borne supérieure de la variable z. Plus précisément, $\mathcal{R}_1$ stipule que la cardinalité de A est une borne inférieure valide pour z. Autrement dit, si k variables ont des domaines deux à deux disjoints, alors il faut au moins k ressources pour réaliser toutes les tâches. Lorsque la cardinalité du stable A est égale à la borne supérieure de z, $\mathcal{R}_2$ stipule que les variables de $\mathcal{X}$ prennent nécessairement leurs valeurs parmi le sous-ensemble des valeurs induit par A. Si tel n'était pas le cas, alors il existerait au moins une variable $x_v \in \mathcal{X}$ qui prendrait une valeur $v \notin \cup_{a \in A} d(x_a)$. Puisque les variables associées à A prennent déjà des valeurs deux à deux différentes (par définition d'un graphe d'intersection et d'un stable), cela signifie qu'il faut au moins $|A \cup v| = \overline{z} + 1$ valeurs différentes pour réaliser toutes les tâches. Autrement dit, $\overline{z} = \overline{z} + 1$, ce qui est impossible.

$$\mathcal{R}_1 : \quad \underline{z} \leftarrow max(\underline{z}, |A|)$$
$$\mathcal{R}_2 : \quad |A| = \overline{z} \Rightarrow \forall i \in V, d(x_i) \leftarrow d(x_i) \cap \bigcup_{a \in A} d(x_a)$$

Pour résumer, le propagateur *greedy* part d'un graphe G, applique une fonction `F` pour obtenir un stable dans G, puis filtre les domaines des variables selon un ensemble de règles de filtrage $\mathcal{R}$. Nous introduisons la notation $\texttt{AMNV}\langle G|\texttt{F}|\mathcal{R}\rangle$ pour définir une famille de propagateur pour la contrainte `atMostNValue`. Avec cette notation, le propagateur *greedy* est alors noté $\texttt{AMNV}\langle G_{\mathcal{I}}|\texttt{MD}|\mathcal{R}_{1,2}\rangle$. Pour illustrer ce propagateur, nous l'appliquons à notre exemple (*cf.* Figure 9.3), dans le cas où z est instancié à la valeur 1. Nous faisons cette hypothèse de manière à déclencher immédiatement le filtrage de la règle $\mathcal{R}_2$. Remarquons qu'il s'agit d'une hypothèse envisageable, même en début de résolution, puisqu'il est possible, par exemple, de brancher sur l'objectif. En raison des domaines des variables, le graphe d'intersection correspondant à notre exemple est un graphe complet. Par conséquent, le stable construit par `MD` ne contient qu'un seul sommet, comme par exemple x_3. $\mathcal{R}_1$ stipule alors que le nombre de valeurs différentes est supérieur ou égal à 1, ce qui n'engendre aucune déduction puisque $d(z) = \{1\}$. $\mathcal{R}_2$ stipule alors que les valeurs $2, 4$ et 5 doivent être retirées du domaine des variables (*cf.* Figure 9.3a). Par conséquent, les arêtes (x_1, x_5) et (x_5, x_4) sont retirées, ce qui conduit à un nouveau graphe d'intersection. À partir de ce nouveau graphe d'intersection, si l'on suppose que `MD` construit le stable composé des sommets x_1 et x_5 (*cf.* Figure 9.3b) alors $\mathcal{R}_1$ stipule que le nombre de valeurs différentes est supérieur ou égal à 2, ce qui conduit à une contradiction ($d(z) = \{1\}$).

En ce qui concerne la complexité temporelle des règles $\mathcal{R}_1$ et $\mathcal{R}_2$, il est évident que $\mathcal{R}_1$ s'effectue



(a) Première propagation (b) Seconde propagation

FIGURE 9.3 – Utilisation de $\texttt{AMNV}\langle G_{\mathcal{I}}|\texttt{MD}|\mathcal{R}_{1,2}\rangle$ sur notre exemple, avec $d(z) = \{1\}$.

en temps constant, ce qui n'est pas le cas de $\mathcal{R}_2$. Par la suite, les notations $n = |\mathcal{X}|$ et $l = |\mathcal{S}|$ désignerons respectivement le nombre de variables et le nombre de valeurs. En supposant que l'union et l'intersection des domaines de deux variables s'obtiennent chacune en $O(\log(l))$, nous obtenons alors une complexité temporelle en $O(n\log(l))$ pour $\mathcal{R}_2$. En effet, l'ensemble $\bigcup_{a\in A} d(x_a)$ est indépendant de i, ce qui signifie qu'il peut être calculé une seule fois.

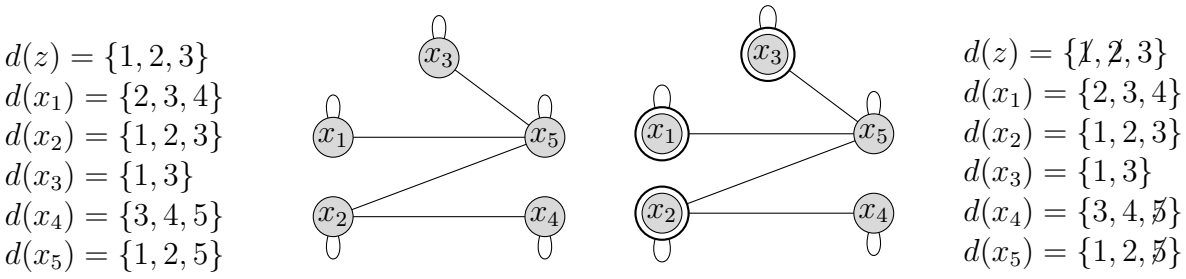
9.2.2 Filtrer `atMostNValue` en présence de contraintes de différence

Puisque le SMPTSP repose uniquement sur deux contraintes globales (`atMostNValue` et `allDifferent`), il pourrait être bien plus efficace de filtrer directement leur conjonction. Nous proposons dans cette optique un propagateur de la famille $\texttt{AMNV}\langle G|\texttt{F}|\mathcal{R}\rangle$, qui tire parti de la structure du problème. Ce propagateur repose sur un constat relativement simple : soit un ensemble de variables $\mathcal{X}$ et deux variables x_i et x_j appartenant à cet ensemble, alors l'existence d'une contrainte de différence entre x_i et x_j contraint ces variables à prendre des valeurs différentes, si bien que pour toute solution, les sommets i et j du graphe d'intersection $G_{\mathcal{I}}(\mathcal{X})$ ne sont jamais adjacents. Cependant, une telle arête peut initialement exister et rester au fil de la résolution. Dans ce cas, les stables construits à partir de ce graphe seront vraisemblablement moins grands qu'ils ne pourraient l'être, conduisant à un filtrage de moindre efficacité. En effet, plus $G_{\mathcal{I}}(\mathcal{X})$ est dense, plus les stables qui y sont construits sont de taille réduite. Autrement dit, il faut retirer ces arêtes le plus en amont possible. Pour cela, nous introduisons la notion de *graphe contraint d'intersection*, et proposons d'utiliser ce nouveau graphe à la place de $G_{\mathcal{I}}(\mathcal{X})$.

Définition 5. *Soit un ensemble de variables $\mathcal{X}$ et un ensemble de contraintes de différence $\mathcal{D}$, le graphe contraint d'intersection de $\mathcal{X}$ et $\mathcal{D}$, noté $G_{\mathcal{CI}}(\mathcal{X}, \mathcal{D}) = (V, E_{\mathcal{CI}})$, est défini par un ensemble de sommets V où chaque sommet $i \in V$ est associé à une variable $x_i \in \mathcal{X}$, et $E_{\mathcal{CI}}$ représente les classes potentielles d'équivalence : pour chaque couple de sommets $(i, j) \in V^2$, il y a une arête $(i, j) \in E_{\mathcal{CI}}$ si et seulement si $d(x_i) \cap d(x_j) \neq \emptyset$ et $\texttt{neq}(x_i, x_j) \notin \mathcal{D}$.*

Pour illustrer $G_{\mathcal{CI}}$, nous l'appliquons à présent sur notre exemple, avec $d(z) = \{1, 2, 3\}$ (*cf.* Figure 9.4). En raison des contraintes de différence, $G_{\mathcal{CI}}$ est moins dense que $G_{\mathcal{I}}$ (*cf.* Figure 9.4a). Par conséquent, `MD` est capable de construire un plus grand stable, conduisant ainsi à une meilleure borne inférieure de z. Par exemple, si nous considérons le stable $\{x_1, x_2, x_3\}$ (*cf.* Figure 9.4b), $\mathcal{R}_1$ et $\mathcal{R}_2$ stipulent respectivement que le nombre de valeurs différentes est d'au moins 3 et que la valeur 5 peut être supprimée des domaines des variables.

Puisque nous considérons ici un seul ensemble de variables $\mathcal{X}$ ainsi qu'un seul ensemble de contraintes $\mathcal{D}$, $G_{\mathcal{CI}}(\mathcal{X}, \mathcal{D})$ et $G_{\mathcal{I}}(\mathcal{X})$ seront respectivement notés $G_{\mathcal{CI}}$ and $G_{\mathcal{I}}$. Par définition, $G_{\mathcal{CI}}$ s'obtient en retirant toutes les arêtes de $G_{\mathcal{I}}$ qui correspondent à des variables liées par une contrainte



(a) Graphe contraint d'intersection

(b) Filtrage à partir d'un stable

FIGURE 9.4 – Utilisation de $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{MD}|\mathcal{R}_{1,2}\rangle$ sur notre exemple

de différence. Par conséquent, $G_{\mathcal{CI}} \subseteq G_{\mathcal{I}}$ et donc, $\mathcal{IS}(G_{\mathcal{I}}) \subseteq \mathcal{IS}(G_{\mathcal{CI}})$, d'où $\alpha(G_{\mathcal{I}}) \leq \alpha(G_{\mathcal{CI}})$. Autrement dit, un stable maximum dans $G_{\mathcal{I}}$ est également un stable dans $G_{\mathcal{CI}}$, mais pas nécessairement maximal. À titre d'exemple, considérons un ensemble non vide de variables avec des domaines identiques ainsi que des contraintes de différence sur chaque paire de variable. Dans ce cas, $G_{\mathcal{I}}$ est un graphe complet alors que $G_{\mathcal{CI}}$ ne contient que des boucles. Par conséquent, $\alpha(G_{\mathcal{I}}) = 1$ alors que $\alpha(G_{\mathcal{CI}}) = |V|$. Étant données les règles de filtrage considérées ($\mathcal{R}_1$ et $\mathcal{R}_2$), il est souhaitable d'obtenir les stables les plus grands possibles et, par conséquent, il est a priori préférable d'utiliser $G_{\mathcal{CI}}$ à la place de $G_{\mathcal{I}}$ (*cf.* Proposition 1).

Afin de comparer le filtrage de deux propagateurs, nous reprenons ici le vocabulaire de Debruyne et Bessière [70] : nous dirons que le propagateur A domine le propagateur B si, pour toute propagation, toute valeur filtrée par B est également filtrée par A ; nous dirons que le propagateur A domine strictement le propagateur B si A domine B et s'il existe au moins un cas particulier pour lequel A filtre une valeur qui n'est pas filtrée par B durant une phase de propagation.

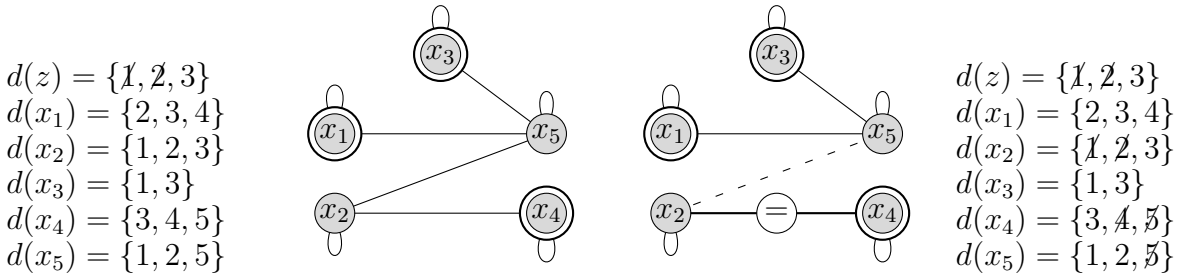
Proposition 1. *Soit un oracle* `O` *capable de trouver tous les stables maximum de n'importe quel graphe, alors* $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{O}|\mathcal{R}_{1,2}\rangle$ *domine strictement* $\texttt{AMNV}\langle G_{\mathcal{I}}|\texttt{O}|\mathcal{R}_{1,2}\rangle$.

Démonstration. Tout d'abord, puisque `O` est capable de trouver tous les stables maximum de n'importe quel graphe, la borne inférieure donnée par $\mathcal{R}_1$ dans $G_{\mathcal{CI}}$ est égale à $\alpha(G_{\mathcal{CI}})$ alors que celle obtenue dans $G_{\mathcal{I}}$ est égale à $\alpha(G_{\mathcal{I}})$. Puisque $\alpha(G_{\mathcal{I}}) \leq \alpha(G_{\mathcal{CI}})$ (par définition de $G_{\mathcal{CI}}$), $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{O}|\mathcal{R}_1\rangle$ domine $\texttt{AMNV}\langle G_{\mathcal{I}}|\texttt{O}|\mathcal{R}_1\rangle$. Ensuite, puisque $\mathcal{IS}(G_{\mathcal{I}}) \subseteq \mathcal{IS}(G_{\mathcal{CI}})$ (par définition de $G_{\mathcal{CI}}$), les valeurs filtrées par $\texttt{AMNV}\langle G_{\mathcal{I}}|\texttt{O}|\mathcal{R}_2\rangle$ sont également filtrées par $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{O}|\mathcal{R}_2\rangle$. Par conséquent, $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{O}|\mathcal{R}_2\rangle$ domine $\texttt{AMNV}\langle G_{\mathcal{I}}|\texttt{O}|\mathcal{R}_2\rangle$. Pour finir, les Figures 9.3 et 9.4 illustrent un cas particulier où $\alpha(G_{\mathcal{CI}}) > \alpha(G_{\mathcal{I}})$. En conséquence, $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{O}|\mathcal{R}_{1,2}\rangle$ domine strictement $\texttt{AMNV}\langle G_{\mathcal{I}}|\texttt{O}|\mathcal{R}_{1,2}\rangle$. □

Proposition 2. *Soit un stable A dans $G_{\mathcal{CI}}$ tel que $|A| = \overline{z}$, toute solution de la conjonction de* `atMostNValue` *et des contraintes de différence $\mathcal{D}$ vérifie la relation suivante : $\forall i \in V \backslash A$, $\exists a \in A_i | x_i = x_a$, avec $A_i = \{a \in A | (i, a) \in E_{\mathcal{CI}}\}$.*

Démonstration. Soit un stable A dans $G_{\mathcal{CI}}$ tel que $|A| = \overline{z}$. Supposons qu'il existe une solution S à la conjonction de la contrainte `atMostNValue` et des contraintes de différence $\mathcal{D}$ telle qu'il existe un sommet $i \in V \backslash A$ pour lequel nous ayons : $\forall a \in A_i, x_i \neq x_a$. Alors, S est également solution de la conjonction de la contrainte `atMostNValue` et des contraintes de différence $\mathcal{D} \cup \{\texttt{neq}(x_i, x_a) | a \in A_i\}$. Par conséquent, $A \cup \{i\}$ est un stable dans $G_{\mathcal{CI}}$, et $\mathcal{R}_1$ stipule alors que $\underline{z} \leftarrow |A \cup \{i\}|$, *i.e.* $\underline{z} \leftarrow \overline{z} + 1$, ce qui est en contradiction avec l'hypothèse de départ. Autrement dit, S n'existe pas, vérifiant ainsi la proposition 2. □

En terme de filtrage, la proposition 2 conduit à la règle $\mathcal{R}_3$, qui stipule que chaque variable x_i prend nécessairement une valeur du sous-ensemble de valeurs induit par l'intersection de l'ensemble A avec



(a) Filtrage de $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{MD}|\mathcal{R}_{1,2}\rangle$ (b) Filtrage de $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{MD}|\mathcal{R}_{1,3}\rangle$

FIGURE 9.5 – $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{MD}|\mathcal{R}_{1,2}\rangle$ comparé à $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{MD}|\mathcal{R}_{1,3}\rangle$, sur notre exemple

le voisinage du sommet i. Autrement dit, $\mathcal{R}_3$ est une variante de $\mathcal{R}_2$. Bien que cette modification soit relativement simple, elle peut avoir un impact important en pratique, notamment sur des problèmes de grande échelle où pour tout sommet $i \in V \backslash A$, le cardinal $|A_i|$ est a priori plus petit que le cardinal $|A|$. Remarquons également le cas particulier qui survient lorsque, pour un sommet donné $i \in V \backslash A$, nous avons $A_i = \{a\}$. Nous sommes alors en mesure de déduire l'égalité $x_i = x_a$. De cette manière, il devient possible de filtrer le domaine des variables correspondant au stable, en plus des autres variables. D'un point de vue théorique, $\mathcal{R}_3$ domine également $\mathcal{R}_2$ (*cf.* Proposition 3). En revanche, $\mathcal{R}_3$ présente une complexité théorique dans le pire des cas en $O(n^2 \log(l))$ contre $O(n \log(l))$ pour $\mathcal{R}_2$. En effet, l'ensemble $\bigcup_{a \in A_i} d(x_a)$ dépend de i et doit donc être calculé à chaque fois.

$$\mathcal{R}_3 : \quad |A| = \overline{z} \Rightarrow \forall i \in V \backslash A \begin{cases} d(x_i) \leftarrow d(x_i) \cap \bigcup_{a \in A_i} d(x_a) \\ A_i = \{a\} \Rightarrow d(x_a) \leftarrow d(x_a) \cap d(x_i) \end{cases}$$

Pour illustrer $\mathcal{R}_3$, nous l'appliquons à présent à notre exemple, avec $d(z) = \{1, 2, 3\}$ (*cf.* Figure 9.5). Si on considère le stable $\{x_1, x_3, x_4\}$, alors $\mathcal{R}_1$ conduit à $d(z) = \{3\}$ mais $\mathcal{R}_2$ ne peut filtrer aucun domaine des variables $\mathcal{X}$ (*cf.* Figure 9.5a), car l'ensemble des valeurs induit par le stable comprend toutes les valeurs possibles $\{1, 2, 3, 4, 5\}$. En revanche, $\mathcal{R}_3$ est capable de filtrer la valeur 5 du domaine de x_5, car elle n'est pas incluse dans l'ensemble $d(x_1) \cup d(x_3) = \{1, 2, 3, 4\}$ (*cf.* Figure 9.5b). $\mathcal{R}_3$ est également capable de filtrer les valeurs 1 et 2 du domaine de x_2, car elles n'apparaisent pas dans l'ensemble $d(x_4) = \{3, 4, 5\}$. Pour finir, $\mathcal{R}_3$ est capable de déduire l'égalité $x_2 = x_4$, qui permet à son tour de filtrer les valeurs 4 et 5 du domaine de x_4.

Proposition 3. *Soit une heuristique déterministe* $\texttt{H}$ *qui construit un stable* A *dans le graphe* $G_{\mathcal{CI}}$*, alors* $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{H}|\mathcal{R}_3\rangle$ *domine strictement* $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{H}|\mathcal{R}_2\rangle$*.*

Démonstration. Puisque $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{H}|\mathcal{R}_3\rangle$ et $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{H}|\mathcal{R}_2\rangle$ utilisent la même heuristique déterministe $\texttt{H}$ dans le même graphe $G_{\mathcal{CI}}$, leur filtrage s'effectue donc à partir du même stable A. Puisque pour tout sommet $i \in V$, nous avons $A_i = \{a \in A | (i, a) \in E_{\mathcal{CI}}\}$, nous obtenons alors $A_i \subseteq A$. Par conséquent, toute valeur filtrée par $\mathcal{R}_2$ est également filtrée par $\mathcal{R}_3$. Autrement dit, $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{H}|\mathcal{R}_3\rangle$ domine $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{H}|\mathcal{R}_2\rangle$. La Figure 9.5 illustre un cas particulier où $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{H}|\mathcal{R}_3\rangle$ filtre plus que $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{H}|\mathcal{R}_2\rangle$. Par conséquent, $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{H}|\mathcal{R}_3\rangle$ domine strictement $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{H}|\mathcal{R}_2\rangle$. □

9.2.3 Diversification du filtrage avec l'algorithme $\texttt{R}^k$

En PPC, le mécanisme de branchement est généralement déclenché après obtention d'un point fixe, *i.e.* un état des domaines des variables où plus aucune contrainte ne peut filtrer la moindre valeur [179]. Dans ce cas, notre modèle serait amené à contruire des milliers de stables durant la phase de résolution. Bessière *et al.* [27] conseillent par conséquent d'utiliser une approche gloutone, telle que `MD`, pour obtenir des stables et filtrer la contrainte `atMostNValue`. L'heuristique `MD` est très

efficace, mais si on suppose que les symétries des sommets de même degré sont retirées de manière lexicographique, alors MD est déterministe et ne permet donc pas de diversifier le filtrage. Comme cela est suggéré par Beldiceanu [22] et Bessière *et al.* [27], et souligné par la proposition 4, il peut être intéressant de chercher plusieurs stables de manière à appliquer les règles de filtrage sur un plus grand nombre de variables.

Proposition 4. *Soient deux fonctions* F_1 *et* F_2 *qui construisent chacune un ensemble de stables dans* $G_{\mathcal{CI}}$*, notés respectivement* $\mathcal{A}_1$ *et* $\mathcal{A}_2$*. Si* F_1 *et* F_2 *sont tels que pour tout* $\mathcal{A}_1$ *induit par* F_1 *et pour tout* $\mathcal{A}_2$ *induit par* F_2*,* $\max_{I_2 \in \mathcal{A}_2 \setminus \mathcal{A}_1} |I_2| < \max_{I_1 \in \mathcal{A}_1} |I_1|$*, alors* $\mathit{AMNV}\langle G_{\mathcal{CI}} | F_1 | \mathcal{R}_{1,3} \rangle$ *domine strictement* $\mathit{AMNV}\langle G_{\mathcal{CI}} | F_2 | \mathcal{R}_{1,3} \rangle$*.*

Démonstration. Tout d'abord, la condition $\max_{I_2 \in \mathcal{A}_2 \setminus \mathcal{A}_1} |I_2| < \max_{I_1 \in \mathcal{A}_1} |I_1|$ implique la relation $\max_{I_2 \in \mathcal{A}_2} |I_2| \leq \max_{I_1 \in \mathcal{A}_1} |I_1|$, et donc $\texttt{AMNV}\langle G_{\mathcal{CI}} | \texttt{F}_1 | \mathcal{R}_1 \rangle$ domine $\texttt{AMNV}\langle G_{\mathcal{CI}} | \texttt{F}_2 | \mathcal{R}_1 \rangle$. Ensuite, supposons à présent qu'une valeur est retirée du domaine d'une variable de $\mathcal{X}$ par $\texttt{AMNV}\langle G_{\mathcal{CI}} | \texttt{F}_2 | \mathcal{R}_3 \rangle$. Si ce filtrage est effectué en considérant un stable de $\mathcal{A}_1 \cap \mathcal{A}_2$, alors le même filtrage est effectué par $\texttt{AMNV}\langle G_{\mathcal{CI}} | \texttt{F}_1 | \mathcal{R}_{1,3} \rangle$. Sinon, ce filtrage est effectué en considérant un stable I_2 de $\mathcal{A}_2 \setminus \mathcal{A}_1$. Une condition nécessaire pour que $\mathcal{R}_3$ puisse filtrer est que l'on ait $|I_2| = \overline{z}$. Puisque $|I_2| < \max_{I_1 \in \mathcal{A}_1} |I_1|$, le problème est alors infaisable, ce qui est capturé par $\texttt{AMNV}\langle G_{\mathcal{CI}} | \texttt{F}_1 | \mathcal{R}_1 \rangle$. Par conséquent, $\texttt{AMNV}\langle G_{\mathcal{CI}} | \texttt{F}_1 | \mathcal{R}_{1,3} \rangle$ domine $\texttt{AMNV}\langle G_{\mathcal{CI}} | \texttt{F}_2 | \mathcal{R}_{1,3} \rangle$. Supposons à présent que $\texttt{F}_1 = \texttt{MD}$ et que $\texttt{F}_2 = \texttt{FV}$, avec FV une heuristique sélectionnant uniquement le premier sommet. Nous supposons que les symétries sont retirées par ordre lexicographique, ce qui signifie par exemple que, dans le cas d'un graphe complet, ces deux heuristiques construisent le même stable, composé alors d'un unique sommet. Si le graphe n'est pas complet, alors il existe au moins un couple de sommets $(u, v) \in G_{\mathcal{CI}}$ n'étant pas adjacents, et donc, n'importe quel stable construit par MD comprendra au moins 2 sommets. Par conséquent, $\texttt{AMNV}\langle G_{\mathcal{CI}} | \texttt{MD} | \mathcal{R}_{1,3} \rangle$ domine strictement $\texttt{AMNV}\langle G_{\mathcal{CI}} | \texttt{FV} | \mathcal{R}_{1,3} \rangle$. Au final, si $\texttt{F}_1$ et $\texttt{F}_2$ sont tels que pour tout $\mathcal{A}_1$ induit par $\texttt{F}_1$ et pour tout $\mathcal{A}_2$ induit par $\texttt{F}_2$, $\max_{I_2 \in \mathcal{A}_2 \setminus \mathcal{A}_1} |I_2| < \max_{I_1 \in \mathcal{A}_1} |I_1|$, alors $\texttt{AMNV}\langle G_{\mathcal{CI}} | \texttt{F}_1 | \mathcal{R}_{1,3} \rangle$ domine strictement $\texttt{AMNV}\langle G_{\mathcal{CI}} | \texttt{F}_2 | \mathcal{R}_{1,3} \rangle$. □

Si tous les stables maximum d'un graphe sont connus, alors $\mathcal{R}_{1,3}$ peut être utilisée de manière optimale, *i.e.* toutes les valeurs pouvant être filtrées par $\mathcal{R}_{1,3}$ sont effectivement filtrées. Une manière d'obtenir tous les stables d'un graphe est d'adapter l'algorithme de Bron-Kerbosch [34] qui permet de trouver toutes les cliques maximales dans un graphe non orienté. Cependant, puisque le problème sous-jacent est NP-Difficile, utiliser l'algorithme de Bron-Kerbosch à chaque phase de propagation serait a priori extrêmement coûteux en temps de calcul. Sur les instances de la littérature, il s'est avéré que l'accroissement du filtrage consécutif à l'utilisation de cet algorithme était beaucoup trop coûteux en temps pour être rentable. Par conséquent, nous suggérons de conserver l'idée initiale de MD et de chercher à obtenir de la diversification au moyen d'algorithmes rapides. Il est tout d'abord envisageable de conserver MD en retirant les symétries de manière aléatoire, mais cela n'engendre pas assez de diversification pour améliorer les résultats. Par conséquent, nous proposons l'algorithme $\texttt{R}^k$ (*cf.* Algorithme 9.1).

L'algorithme $\texttt{R}^k$ effectue k itérations, chacune d'elles construisant un stable de manière aléatoire. Chaque stable s'obtient en itérant entre deux phases consistant à sélectionner aléatoirement un sommet puis à retirer ce sommet, tous ses voisins et les arêtes correspondantes. On obtient ainsi un ensemble de k stables, chacun étant maximal du point de vue de l'inclusion. D'un point de vue théorique, cette approche présente plusieurs propriétés intéressantes. Elle offre tout d'abord un moyen de contrôler la complexité dans le pire des cas ainsi que la puissance du filtrage. De plus, le fait de construire les stables de manière aléatoire tend à impacter les domaines des variables de manière homogène.

Algorithme 9.1: Algorithme $\mathtt{R}^k$ permettant de construire k stables

Données :
$G_{\mathcal{CI}}$, le graphe contraint d'intersection
G, un graphe
k, le nombre d'itérations

```
1  A ← ∅
2  pour (compteur = 1..k) faire
3      G ← copier(G_CI)
4      A ← ∅
5      tant que (G ≠ ∅) faire
6          x ← selectionnerAleatoirementSommet(G)
7          A ← A ∪ {x}
8          G ← G \ {y|(x, y) ∈ G}
9      A ← A ∪ {A}
10 retourner A
```

9.3 Une stratégie de branchement pour le SMPTSP

9.3.1 Un branchement adapté aux règles de filtrage

En général, les solveurs PPC obtiennent l'optimum d'un problème en énumérant les solutions améliorantes. Par conséquent, lorsque la recherche est complète, la dernière solution trouvée est optimale. On parle alors de stratégie *top-down*. Rappelons que le filtrage de $\texttt{AMNV}\langle G|\texttt{F}|\mathcal{R}\rangle$ est lié à la borne supérieure du problème. Autrement dit, plus la borne supérieure est petite, plus le filtrage est puissant. Par conséquent, nous optons naturellement pour une stratégie *bottom-up*. Il s'agit de chercher une solution de valeur k, où k est initialisé à la borne inférieure de z et incrémenté à chaque fois que le solveur prouve l'infaisabilité. La première solution trouvée est donc optimale. Il suffit pour cela de brancher sur l'objectif et de lui assigner sa borne inférieure.

La stratégie de branchement utilisée sur les variables $\mathcal{X}$ consiste à sélectionner la variable de plus petit domaine [106], et de l'instancier à la première valeur possible qui est déjà utilisée dans la solution courante. Si une telle valeur n'existe pas, alors la variable est instanciée à sa borne inférieure. Ce branchement est de plus renforcé par l'heuristique `last-conflict` [136] qui vise à identifier les variables critiques.

9.3.2 Brancher, c'est parier...

Rappelons tout d'abord que, dans le cas où toutes les ressources sont identiques, le problème peut alors être résolu en temps polynomial. Dans un tel contexte, il serait possible de choisir arbitrairement n'importe quel ensemble de z valeurs et être assuré d'obtenir une solution. En particulier, il serait possible de choisir les z premières valeurs. En supposant que ces valeurs sont consécutives et commencent à 1, nous pourrions alors utiliser la contrainte $\texttt{maximum}(\mathcal{X}) = z$ pour réaliser cette sélection. Une telle contrainte permettrait de couper une large partie de l'arbre de recherche [93]. Bien sûr, dans notre contexte, les ressources ne sont a priori pas identiques, ce qui signifie qu'il n'est pas possible d'utiliser directement la contrainte $\texttt{maximum}(\mathcal{X}) = z$. Néanmoins, il est possible que les ressources soient presque identiques. Dans ce cas, la difficulté principale n'est pas tant de trouver un sous-ensemble de ressources permettant de couvrir toutes les tâches, mais plutôt de trouver l'affectation des tâches à un sous-ensemble de ressources. Autrement dit, en notant z^* la valeur

$d(b_{reif}) = \{0, 1\}$
$d(z) = \{3\}$
$d(x_1) = \{2, 3, 4\}$
$d(x_2) = \{1, 2, 3\}$
$d(x_3) = \{1, 3\}$
$d(x_4) = \{3, 4, 5\}$
$d(x_5) = \{1, 2, 5\}$

(a) Domaines avant de brancher sur b_{reif}

$d(b_{reif}) = \{\emptyset, 1\}$
$d(z) = \{3\}$
$d(x_1) = \{2, 3, \not{4}\}$
$d(x_2) = \{1, 2, 3\}$
$d(x_3) = \{1, 3\}$
$d(x_4) = \{3, \not{4}, \not{5}\}$
$d(x_5) = \{1, 2, \not{5}\}$

(b) Domaines après instanciation de b_{reif} à 1

FIGURE 9.6 – Impact du branchement b_{reif}

optimale de z, il est alors possible de trouver une solution pour un grand nombre d'ensembles de ressources de taille z^*. Pour bénéficier de la contrainte $\texttt{maximum}(\mathcal{X}) = z$ tout en conservant une approche exacte, nous introduisons une variable binaire b_{reif} permettant de réifier cette contrainte [23]. L'heuristique de branchement instancie tout d'abord b_{reif} à 1, ce qui permet de bénéficier du filtrage de la contrainte et de réduire fortement l'arbre de recherche [85]. Si le solveur prouve l'infaisabilité de cette branche, le mécanisme de backtrack conduit alors à instancier b_{reif} à 0, ce qui revient dans le cas d'une réification complète à poster l'opposée de la contrainte initiale, *i.e.* $\texttt{maximum}(\mathcal{X}) \neq z$. On conserve ainsi une approche exacte, capable de gérer efficacement le cas de ressources presque identiques. Bien entendu, la contrainte $\texttt{maximum}(\mathcal{X}) \neq z$ induit un filtrage très faible. Cependant, l'objectif de ce mécanisme n'est pas de mieux filtrer dans la branche de droite, mais de mieux filtrer dans la branche de gauche, tout en conservant une approche exacte. Remarquons que l'ensemble initial des ressources peut être mélangé aléatoirement de manière à gérer le cas où les ressources sont ordonnées d'une manière spécifique. Ce mécanisme de branchement revient en réalité à parier sur l'ensemble des ressources pouvant conduire à une solution optimale. Plus précisément, il existe $\binom{|S|}{z^*}$ ensemble de valeurs dont le cardinal est égal à la valeur optimale. Notons K le nombre d'ensembles différents appartenant à au moins une solution optimale. Nous avons alors $0 \leq K \leq \binom{|S|}{z^*}$. La probabilité que l'ensemble $\{1, 2, ..., z^*\}$ appartienne à une solution optimale est alors $P(b_{reif} = 1) = \frac{K}{\binom{|S|}{z^*}}$.

Pour illustrer l'utilisation de b_{reif}, nous l'appliquons à présent sur notre exemple avec $d(z) = \{3\}$ (*cf.* Figure 9.6). Par souci de clarté, les ressources conservent ici leur ordre initial de manière à ce que la valeur $i \in [1, 5]$ corresponde bien à la ressource s_i. Cependant, en pratique, nous mélangons les ressources de manière aléatoire, si bien que les valeurs 1, 2 et 3 peuvent correspondre par exemple aux ressources s_2, s_4 et s_5. Lorsque b_{reif} est instanciée à 1, la contrainte $\texttt{maximum}(\mathcal{X}) = z$ conduit à retirer les valeurs 4 et 5 des domaines des variables $\mathcal{X}$. Il s'agit ici d'un pari gagné, car il existe une solution optimale utilisant les valeurs $\{1, 2, 3\}$. En tout, seuls deux ensembles de valeurs différents permettent d'obtenir une solution optimale, *i.e.* $K = 2$. Par conséquent, la contrainte $\texttt{maximum}(\mathcal{X}) = z$ a une probabilité d'être vérifiée sur les solutions optimales de $\frac{2}{\binom{5}{3}} = 0.2$, ce qui illustre l'intérêt d'utiliser une contrainte réifiée.

9.4 Résultats expérimentaux

Pour évaluer l'intérêt de notre approche, nous avons réalisé de nombreux tests expérimentaux[1] sur les instances de la littérature, mais également sur un nouveau jeu d'instances. Nous étudions ensuite la

[1] Nos algorithmes sont implémentés en JAVA et nous utilisons Choco 3-1-1 [62] pour implémenter *CP model* ainsi que Cplex 12.4 [114] avec la configuration par défaut pour implémenter *MIP model*. Les tests sont réalisés sur un Intel Core i3-540 (3.06 GHz & 8G RAM).

Instances	Attribut	Valeurs
Data_100	Nombre de tâches	$[\![70, 1600]\!]$
	Nombre de ressources	$[\![60, 950]\!]$
	Disponibilité des ressources	[06h, 14h], [08h, 16h] [14h, 22h], [16h, 24h] [22h, 06h], [24h, 08h]
	Probabilité qu'une ressource puisse être affectée à une tâche	25%
Data_137	Nombre de tâches	$[\![40, 2000]\!]$
	Nombre de ressources	$[\![20, 250]\!]$
	Disponibilité des ressources	[00h, 24h]
	Probabilité qu'une ressource puisse être affectée à une tâche	33%

TABLE 9.1 – Caractéristiques principales des instances du SMPTSP.

qualité de la borne inférieure de l'objectif au nœud racine, puis nous évaluons l'intérêt des procédures de diversification et d'intensification du filtrage proposées précédemment. Nous analysons également l'efficacité générale de l'approche, en évaluant la pertinence de notre stratégie de branchement, ainsi que la capacité de l'approche à borner rapidement et efficacement l'objectif. Pour finir, nous comparons notre approche aux meilleures méthodes de la littérature dédiée au SMPTSP.

Dans toute cette section, nous notons respectivement z^* et $\underline{z}_r$ la valeur optimale de l'objectif et la borne inférieure de l'objectif au nœud racine. De plus, dans le cas d'une stratégie du type *top-down*, l'utilisation du propagateur $\texttt{AMNV}\langle G|\texttt{F}|\mathcal{R}\rangle$, au sein de *CP model* est notée $\downarrow\texttt{AMNV}\langle G|\texttt{F}|R\rangle$ alors que dans le cas d'une approche *bottom-up* nous employons la notation $\uparrow\texttt{AMNV}\langle G|\texttt{F}|\mathcal{R}\rangle$.

9.4.1 Un nouveau jeu d'instances

Smet *et al.* [184] ont récemment montré que les 137 instances de la littérature proposées par Krishnamoorthy *et al.* [128] admettent toutes une solution réalisable avec une valeur de l'objectif égale à M//. Puisque chaque ressource ne peut réaliser qu'un sous-ensemble de tâches, ce résultat est quelque peu surprenant. On s'attendrait en effet à ce que la prise en compte des compétences ait un impact sur la valeur de l'optimum, mais en réalité cela ne fait que compliquer la recherche d'une solution. Puisqu'il n'est pas difficile d'obtenir une excellente borne inférieure sur ces instances, la recherche d'une bonne borne inférieure sur ces instances via notre modélisation PPC n'a pas vraiment de sens. Par conséquent, nous introduisons un nouveau jeu d'instances pour lesquelles la borne inférieure M// ne correspond pas à l'optimum du problème. Les caractéristiques principales des instances générées et des instances de la littérature sont récapitulées à la Table 9.1.

Les nouvelles instances ont été générées via une procédure dédiée, fondée sur plusieurs observations empiriques de la littérature. Tout d'abord, Krishnamoorthy *et al.* [128] mentionnent que le rapport moyen de la somme des durées des tâches sur la somme des durées des fenêtres horaires de disponibilité des ressources doit être proche de 90% pour obtenir des instances difficiles à résoudre. Smet *et al.* [184] expliquent quant à eux que les instances sont d'autant plus difficiles à résoudre que leur durée moyenne est faible. À partir de ces deux conclusions, nous proposons une procédure capable de générer de nouvelles instances difficiles à résoudre. Cette procédure génère aléatoirement un ensemble de tâches avec une durée variant de 15 minutes à 2 heures. Nous considérons des compétences attribuées aléatoirement ainsi que 6 fenêtres horaires de disponibilité pour les ressources. Nous utilisons tout d'abord trois fenêtres horaires de 8 heures de manière à couvrir entièrement une

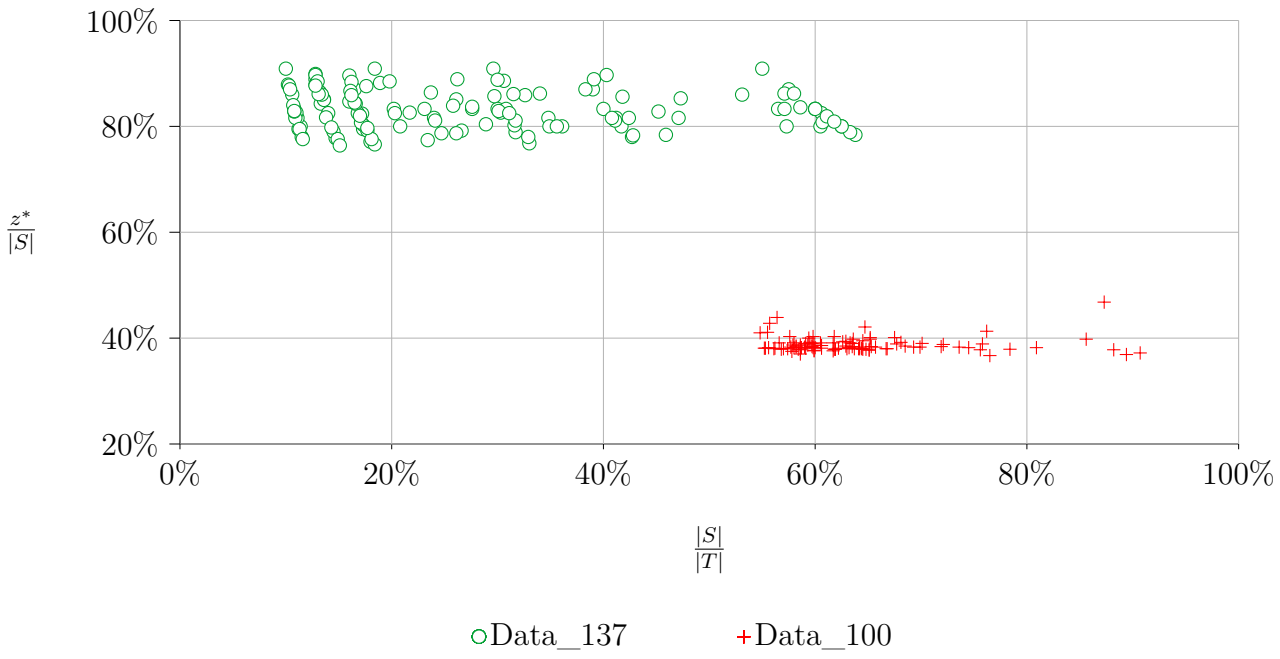


FIGURE 9.7 – Comparaison des instances sur la base de plusieurs critères.

journée de 24 heures, ce qui est habituel en planification de personnel [189], puis nous complétons cet ensemble en ajoutant 3 nouvelles fenêtres horaires, obtenues à partir des précédentes en décalant leur date de début de deux heures, de manière à ce que chaque tâche soit complètement incluse dans l'une de ces fenêtres horaires. À partir de cette procédure nous proposons 100 nouvelles instances avec un nombre de tâches et de ressources variant respectivement de 70 à 1600 et de 60 à 950.

Les instances de Krishnamoorthy *et al.* seront notées Data_137, alors que celles que nous proposons sont notées Data_100. De manière à mieux cerner les différences entre ces deux jeux de données, nous réalisons à présent quelques comparaisons sur ces jeux de données. Nous comparons tout d'abord les ratios $|S|/|T|$ et $z^*/|T|$ sur chaque instance (*cf.* Figure 9.7). Rappelons que $|S|$ et $|T|$ sont des données d'entrée alors que z^* est connu ou estimé après résolution. Nous constatons que les deux jeux de données forment des nuages de points distincts. La différence principale entre les deux jeux de données se situe au niveau du ratio $z^*/|T|$, dont la moyenne varie autour de 39% pour Data_100 contre 83% pour Data_137. En ce qui concerne les données d'entrée uniquement, le ratio $|S|/|T|$ est $2,2$ fois plus grand sur Data_100 que sur Data_137. Autrement dit, les instances de Data_100 ont en moyenne un plus grand nombre de ressources disponibles par tâche que sur Data_137, mais elles utilisent un plus faible pourcentage de ces ressources. En conséquence, il pourrait être plus difficile de trouver un ensemble optimal de ressources sur Data_100 que sur Data_137.

Nous investiguons à présent l'homogénéité des ressources. Pour cela, nous utilisons une distance normalisée de Hamming d_{H}. Plus formellement, la distance $d_{\mathrm{H}}(s_1, s_2)$ entre deux ressources s_1 et s_2 est égale au nombre de tâches appartenant à une seule des deux ressources, divisé par le nombre total de tâches dans l'instance. Nous calculons alors cette distance pour chaque couple de ressource et pour chaque instance. Les résultats obtenus sont analysés au moyen de quartiles, comme cela est illustré à la Figure 9.8. La distance médiane entre deux ressources est en moyenne relativement similaire sur les deux jeux d'instances (44.5% sur Data_137 contre 46.6% sur Data_100). En revanche, le premier et le troisième quartile sont en moyenne très différents sur les deux jeux de données. Plus précisément, sur Data_100, la plupart des ressources sont soit très similaires, soit très différentes les unes des autres, alors que sur Data_137, la quasi totalité des ressources sont toujours un peu différentes, mais presque jamais complètement différentes ou identiques. Cette différence provient sans

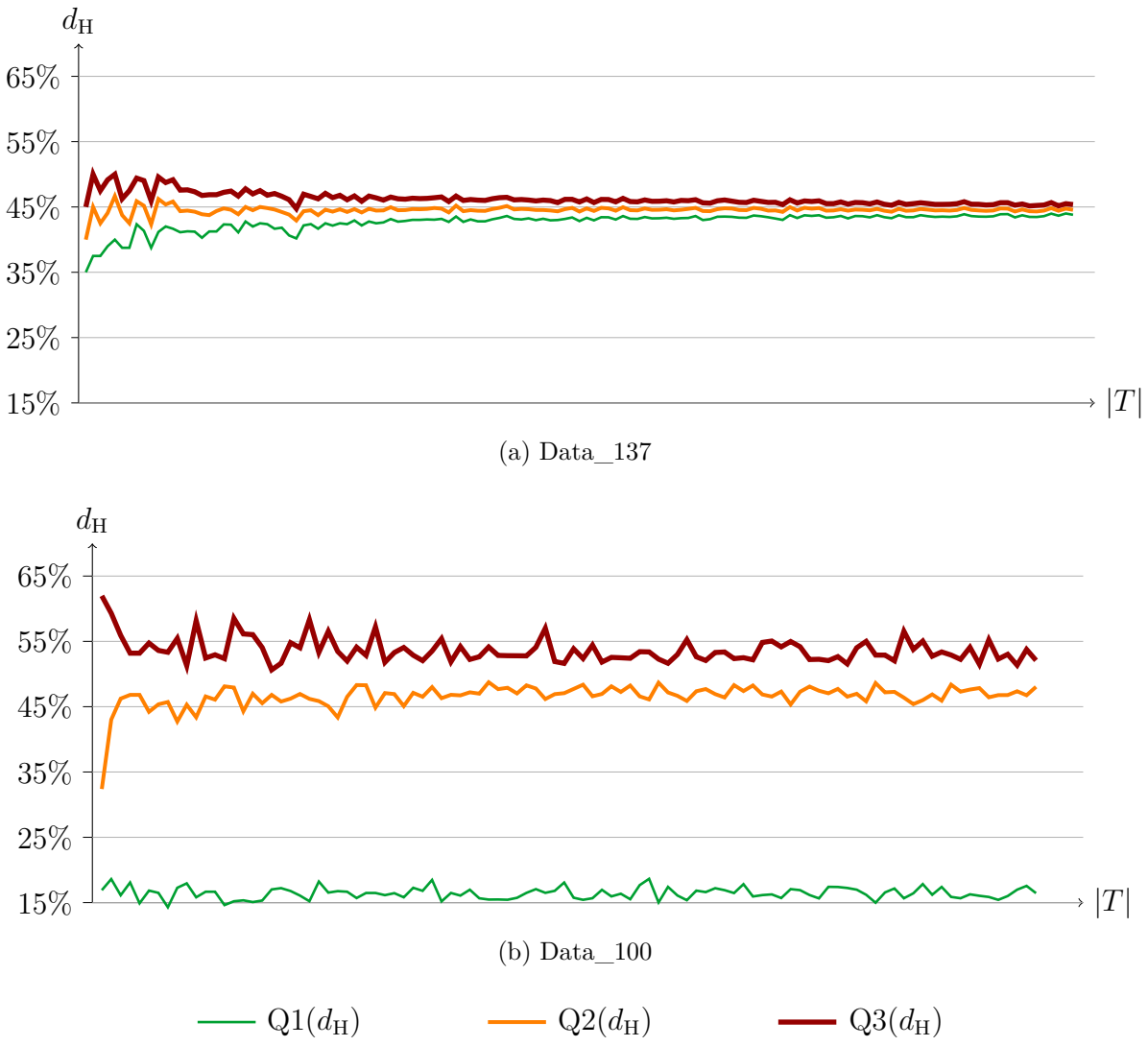


FIGURE 9.8 – Homogénéité des ressources selon les instances triées par taille croissante.

doute des fenêtres horaires de 8 heures utilisées pour Data_100 : les ressources qui appartiennent à la même fenêtre horaire sont très similaires, alors que des ressources appartenant à des fenêtres horaires différentes sont très différentes.

9.4.2 Intérêt du graphe contraint d'intersection $G_{\mathcal{CI}}$

Utiliser $G_{\mathcal{CI}}$ à la place de $G_{\mathcal{I}}$ revient en réalité à réduire le nombre d'arêtes de manière à obtenir des stables plus grands et par voie de conséquence, un filtrage plus efficace. Pour évaluer l'impact de $G_{\mathcal{CI}}$, nous mesurons tout d'abord le ratio $|E_{\mathcal{CI}}|/|E_{\mathcal{I}}|$. Sur Data_137, ce ratio varie de 35% à 85% avec une valeur moyenne de 65%. Sur Data_100, la tendance est différente puisque le ratio varie de 67% à 75%, avec une valeur moyenne de 71%. Ces différences proviennent sans doute de la structure des instances de Data_100 qui est plus marquée que celle des instances de Data_137. Globalement, $G_{\mathcal{CI}}$ a bien moins d'arêtes que que $G_{\mathcal{I}}$.

Nous comparons à présent la valeur de $\underline{z}_r$, obtenue avec $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{MD}|\mathcal{R}_1\rangle$ et celle obtenue avec $\texttt{AMNV}\langle G_{\mathcal{I}}|\texttt{MD}|\mathcal{R}_1\rangle$, sur les deux jeux de données. M// sert ici de point de référence. Les résultats sont illustrés par la Figure 9.10. L'axe des abscisses représente les instances triées par valeur croissante de M//. Sur chaque jeu de données, l'utilisation de $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{MD}|\mathcal{R}_1\rangle$ à la place de $\texttt{AMNV}\langle G_{\mathcal{I}}|\texttt{MD}|\mathcal{R}_1\rangle$ améliore drastiquement la valeur de $\underline{z}_r$. Plus précisément, sur Data_137, $G_{\mathcal{CI}}$ permet d'atteindre un

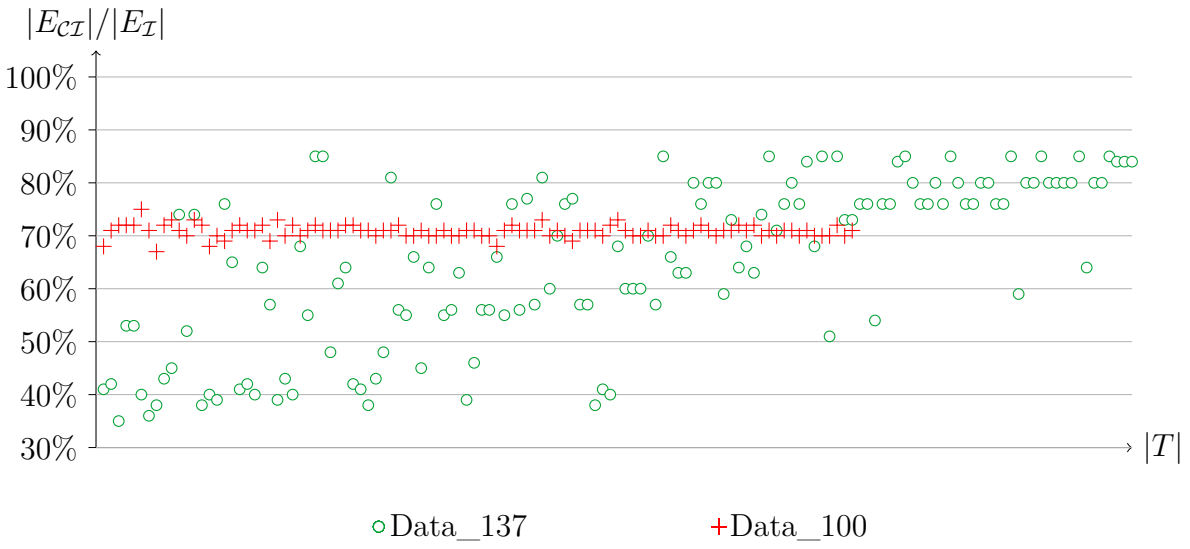


FIGURE 9.9 – Ratio $|E_{\mathcal{CI}}|/|E_{\mathcal{I}}|$ des instances triées par nombre croissant de tâches

écart relatif moyen entre M// et $\underline{z}_r$ d'environ 6%. Puisque M// est toujours égale à z^* sur Data_137, cela signifie que $\underline{z}_r$ est très proche de z^*. Sur Data_100, $G_{\mathcal{CI}}$ permet d'obtenir des valeurs de $\underline{z}_r$ plus de deux fois supérieures à M//. De plus, sur les deux jeux de données, notre approche passe bien mieux à l'échelle que l'approche classique qui n'est pas en mesure d'évaluer correctement $\underline{z}_r$ quelle que soit la taille de l'instance. Plus précisément, la borne inférieure donnée par $\texttt{AMNV}\langle G_{\mathcal{I}}|\texttt{MD}|\mathcal{R}_1\rangle$ tourne autour de 1 (respectivement 6), ce qui correspond au nombre de fenêtres horaires différentes sur Data_137 (respectivement Data_100). En réalité, cela n'est pas surprenant puisque $\texttt{AMNV}\langle G_{\mathcal{I}}|\texttt{MD}|\mathcal{R}_1\rangle$ est complètement aveugle aux contraintes `allDifferent`, qui représentent une part importante du problème. L'utilisation de $G_{\mathcal{CI}}$ à la place de $G_{\mathcal{I}}$ permet à la contrainte `atMostNValue` de voir ces contraintes `allDifferent`, améliorant ainsi l'évaluation de $\underline{z}_r$.

9.4.3 Filtrage, temps de calcul et branchement

Comme cela est expliqué à la Section 9.2.3, l'utilisation de l'algorithme $\texttt{R}^k$ est une manière simple et efficace d'obtenir de la diversification dans le filtrage. En ce qui concerne la distribution de probabilité de $\texttt{R}^k$, deux options naturelles sont envisageables. Il est tout d'abord possible d'utiliser une distribution uniforme, mais il est également possible d'utiliser une distribution pondérée favorisant la sélection de sommets de plus petit degré. Cette seconde option correspond à une variante aléatoire de `MD`. De manière à profiter au maximum de la diversification du filtrage, nous utilisons une distribution uniforme. Le paramètre k permet de gérer le compromis entre le temps de calcul et la qualité du filtrage. Nous avons testé de nombreuses valeurs de k et de nombreuses configurations du propagateur de la contrainte `atMostNValue`. Nous présentons à présent tous ces résultats.

Impact des modifications des règles de filtrage

Pour évaluer l'influence des différentes règles de filtrage sur les performances de `AMNV`, nous comparons le nombre d'optima prouvés par *CP model* avec les règles de filtrage $\mathcal{R}_1$, $\mathcal{R}_{1,2}$ et $\mathcal{R}_{1,3}$, mais sans le branchement sur b_{reif} pour ne pas perturber l'analyse des résultats. (*cf.* Figure 9.11). Sur les deux jeux de données, $\mathcal{R}_{1,2}$ n'est pas beaucoup plus efficace que $\mathcal{R}_1$ alors que cette règle est plus coûteuse en temps de calcul. Autrement dit, $\mathcal{R}_1$ est à elle seule relativement efficace. À l'inverse, $\mathcal{R}_3$ est très efficace sur Data_137, mais beaucoup moins sur Data_100. Remarquons qu'un nombre minimum d'itérations est requis pour que $\mathcal{R}_3$ soit pleinement efficace. Globalement, $\mathcal{R}_3$ conduit à de meilleurs résultats que $\mathcal{R}_{1,2}$, ou bien à des résultats similaires. De manière surprenante, les meilleures configu-

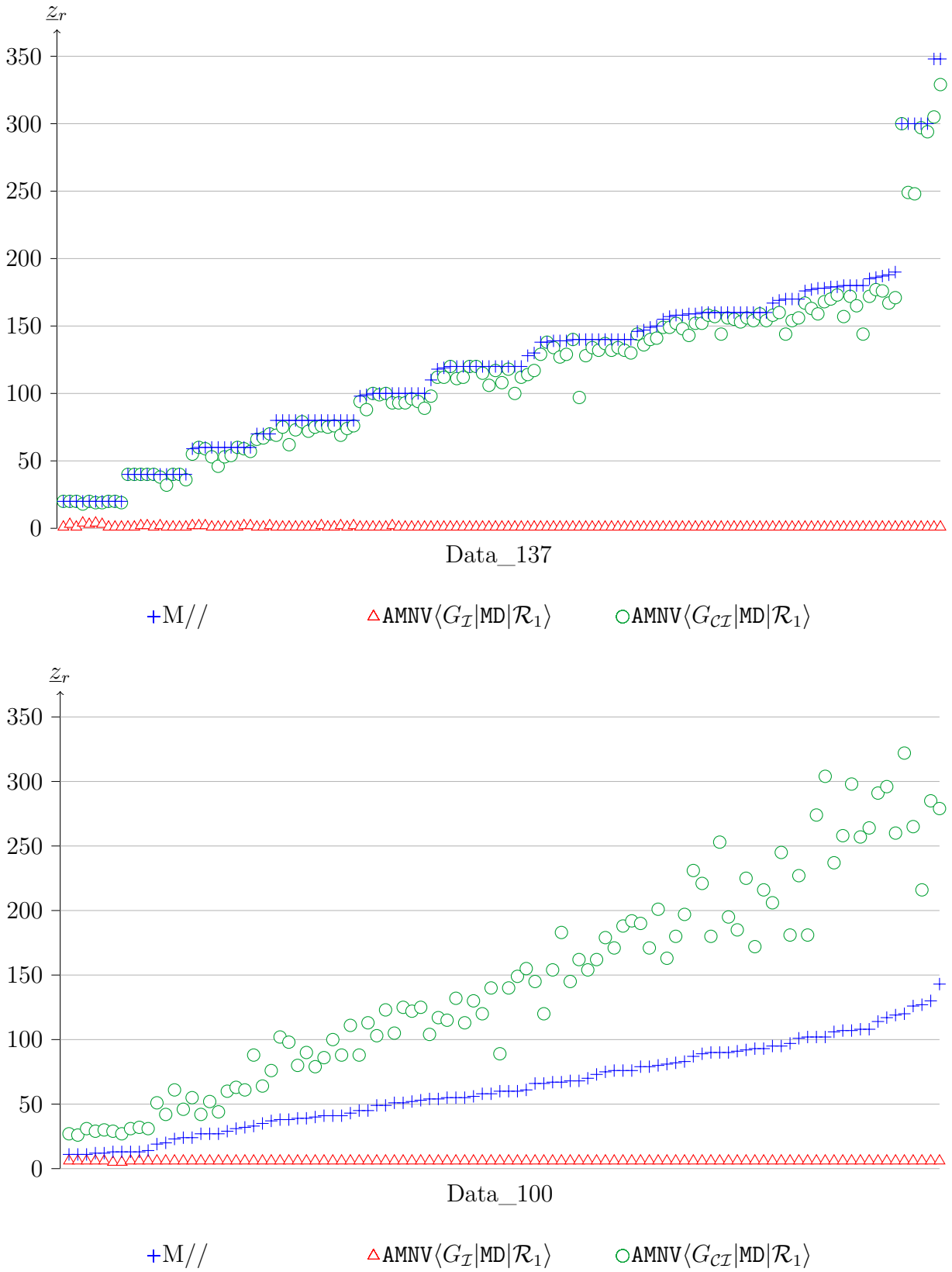
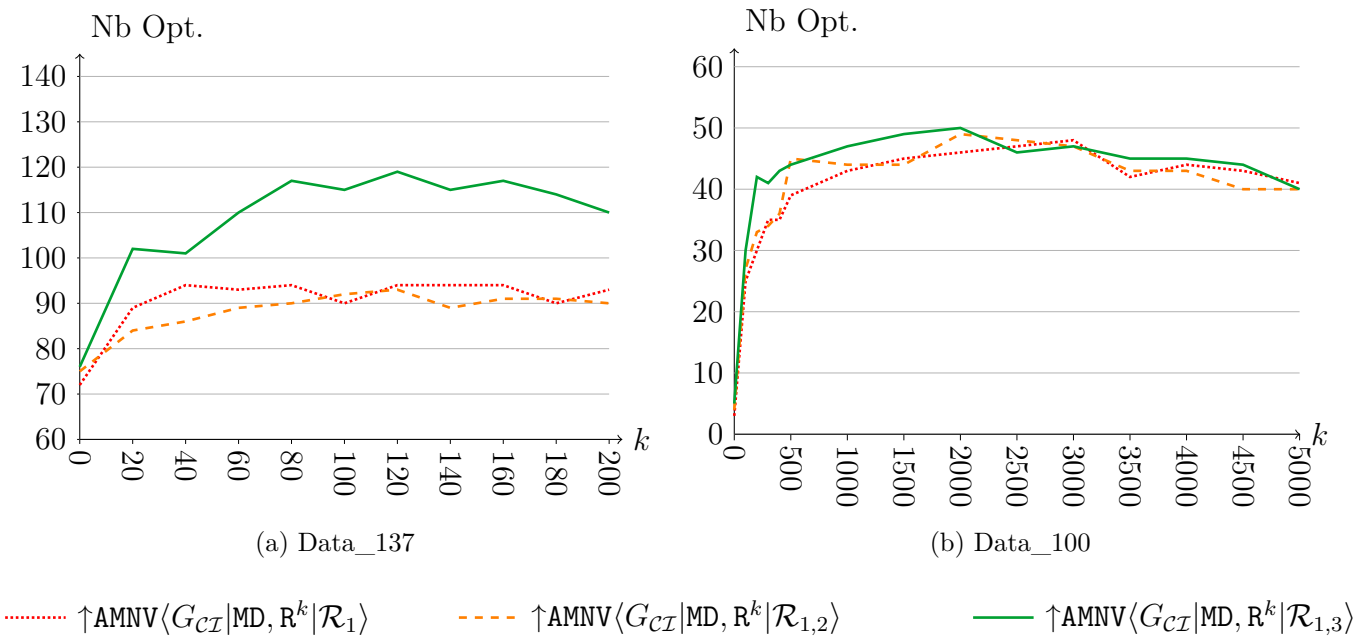


FIGURE 9.10 – Impact de $G_{\mathcal{CI}}$ sur $\underline{z}_r$. Les instances sont triées par M// croissant.



(a) Data_137 (b) Data_100

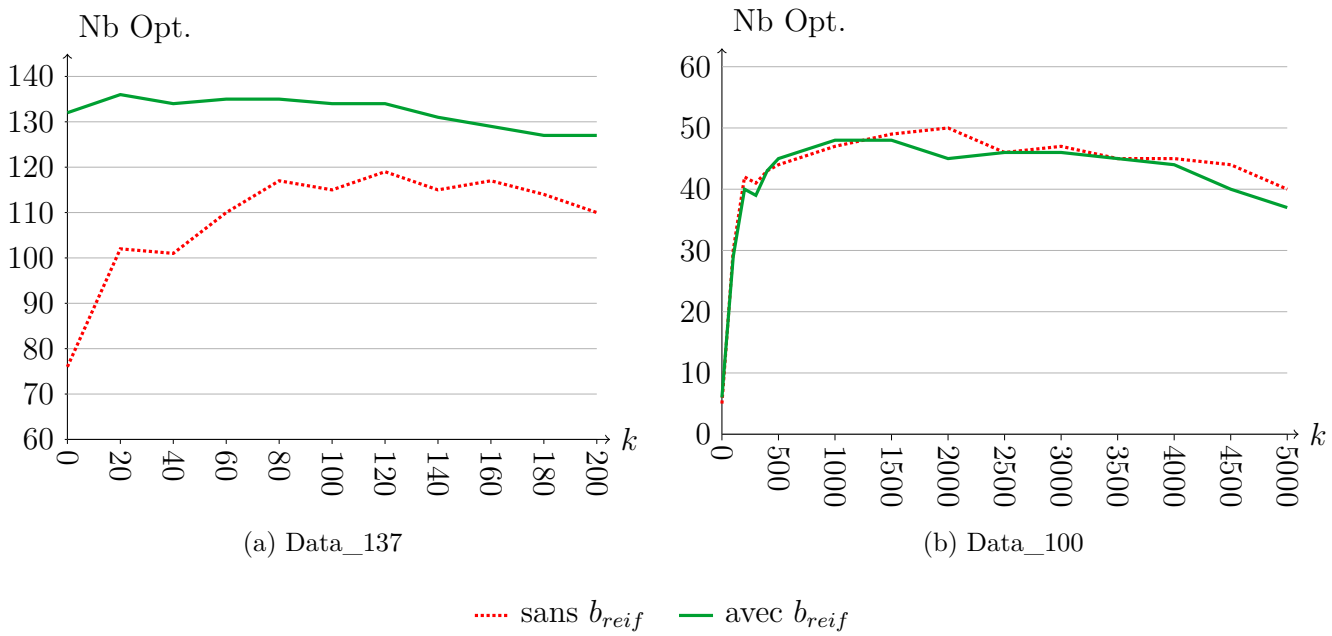
FIGURE 9.11 – Nombre d'optima prouvés après 5 min, en fonction de k, pour plusieurs configurations.

rations pour k changent complètement d'un jeu de données à l'autre. En ce qui concerne Data_137, quelques dizaines d'itérations suffisent pour améliorer les performances, avec une configuration optimale de $k = 120$. A contrario, des milliers d'itérations sont nécessaires sur Data_100. Remarquons en particulier qu'il reste préférable de réaliser 5 000 itérations sur Data_100 plutôt qu'aucune. Autrement dit, R^k permet des déductions qui valent la peine de passer beaucoup de temps à construire un grand nombre de stables. La configuration optimale sur Data_100 tourne autour de 2 000 itérations. En 5 minutes, la meilleure configuration de *CP model* est en mesure de résoudre environ 80% des instances de Data_137 ($k = 120$), comparé à seulement une petite moitié sur Data_100 ($k = 2000$). De manière à conserver un temps de calcul raisonable, nous suggérons une initialisation par défaut entre 20 et 100, ce qui semble plus efficace en moyenne que `MD`, sans pour autant que le temps de calcul n'augmente fortement.

Nous évaluons à présent l'intérêt du branchement sur b_{reif} (*cf.* Figure 9.12). Les résultats montrent que b_{reif} est très efficace sur Data_137. Ce branchement permet en effet de résoudre 136 instances en moins de 5 minutes pour $k = 20$. En revanche, b_{reif} est loin d'être aussi efficace sur Data_100, il entraîne au contraire une légère perte d'efficacité.

9.4.4 Passage à l'échelle

Nous évaluons à présent la capacité de notre approche à fournir de bonnes bornes même sur des instances de grande taille. Pour cela, nous évaluons l'écart relatif $(\overline{z}\text{-}\underline{z})/\overline{z}$ en fonction du modèle utilisé au bout de 6 minutes. Les valeurs par défaut de $\underline{z}$ et $\overline{z}$ sont respectivement de 0 et $|S|$, *i.e.* M// n'est pas utilisée. Par conséquent, lorsqu'une résolution échoue à fournir la moindre borne, l'écart relatif est alors de 100%. Dans le cas de *CP model*, le temps de calcul est réparti équitablement entre les approches *top-down* et *bottom-up*. Étant donnée la taille des instances considérées, il peut être difficile de trouver une affectation réalisable. Puisque la propagation de `atMostNValue` est déclenchée quand la borne supérieure de z est atteinte, nous pouvons espérer une propagation forte avec l'approche *bottom-up*, ce qui n'est plus le cas avec une approche *top-down*. Dans le cas d'une approche *top-down*, il est a priori préférable de rechercher une propagation plus rapide. Par conséquent, le filtrage de ↓AMNV⟨$G_{\mathcal{CI}}$|MD, R^k|$\mathcal{R}_{1,3}$⟩ est allégé en posant $k = 0$ et $b_{reif} = 0$.



(a) Data_137 (b) Data_100

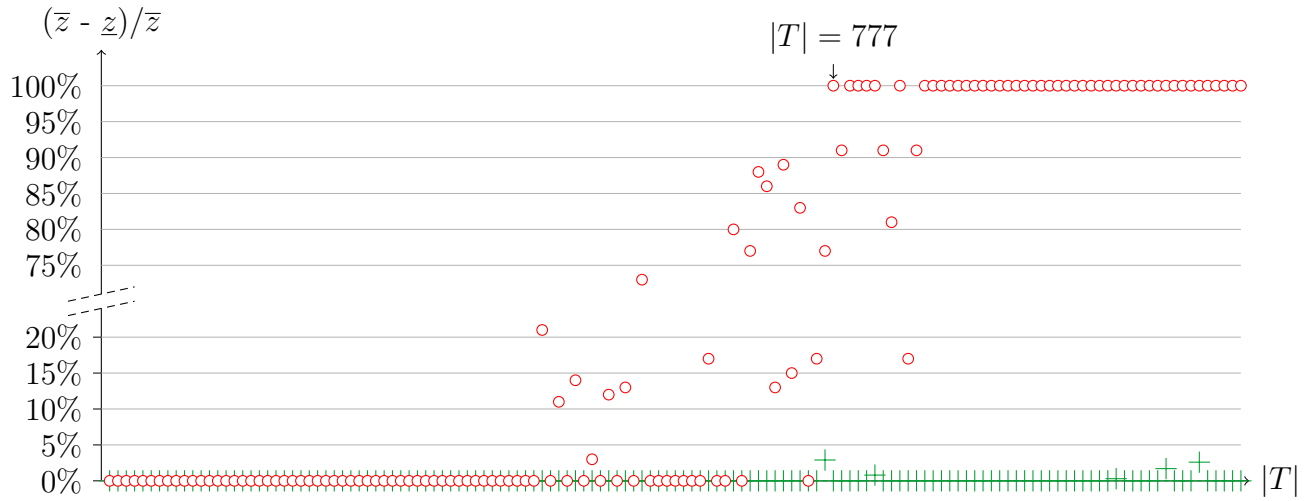
FIGURE 9.12 – Nombre d'optima prouvés après 5 min par $\uparrow\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{MD},\texttt{R}^k|\mathcal{R}_{1,3}\rangle$, selon k et b_{reif}.

La Figure 9.13 illustre la capacité de passage à l'échelle des modèles *MIP model* et *CP model*. Il est clair que *MIP model* échoue à fournir la moindre borne sur la moitié des instances pour les deux jeux de données. Sur Data_137, *MIP model* trouve une borne inférieure mais pas de borne supérieure pour 12 instances. À l'inverse, l'association des approches *top-down* et *bottom-up* permet à *CP model* d'obtenir de très bonnes bornes sur les deux jeux de données. Les écarts relatifs moyens des bornes obtenues par *CP model* sont respectivement de 0.1% et 1.4%, avec des maxima à 2.9% et 5.7% pour Data_137 et Data_100. Autrement dit, l'écart relatif des bornes augmente peu.

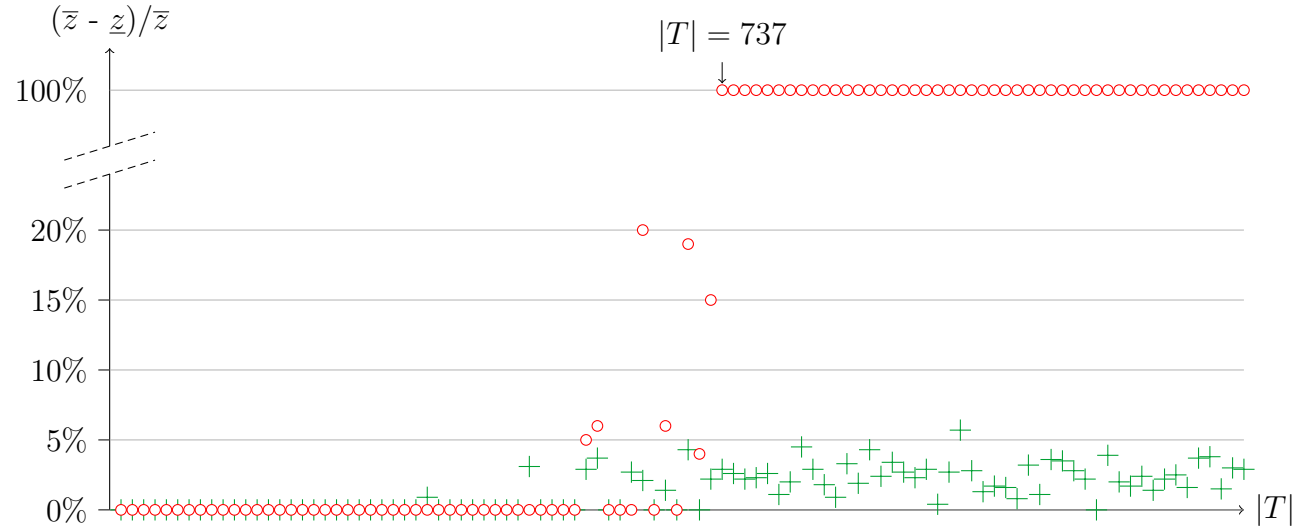
9.4.5 Comparaison à la littérature

Pour évaluer la qualité de notre approche, nous la comparons à présent aux méthodes de la littérature. Les deux premières approches sont données par Krishnamoorthy *et al.* [127], qui introduisent à la fois *MIP model* ainsi qu'une métaheuristique dédiée. Par la suite, Smet *et al.* [184] ont proposé une métaheuristique fondée sur la résolution successive et optimale d'un sous-problème, alors que Lin et Ying [140] proposent une approche exacte en trois phases : la première phase construit une solution initiale, qui est ensuite améliorée avec un algorithme glouton et, pour finir, *MIP model* est initialisé avec cette solution et résout le problème complet. Nous comparons les performances de *CP model* avec ces différentes approches. Pour *CP model*, nous utilisons $\uparrow\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{MD},\texttt{R}^{40}|\mathcal{R}_{1,3}\rangle$, qui est la meilleure configuration sur Data_137. A propos des résultats de Lin et Ying [140], il arrive que leur troisième phase échoue, alors que les phases 1 et 2 ont permis d'obtenir une solution optimale. Il est probable que les auteurs n'étaient pas conscients que M// donne l'optimal sur toutes les instances, ce qui a été démontré en premier par Smet *et al.*. Par conséquent, nous considérons que leur méthode s'arrête dès que l'une des trois phases trouve une solution utilisant M// ressources.

Comme cela est illustré à la Figure 9.14, *CP model* ainsi que la métaheuristique de Smet *et al.* [184] dominent les autres approches. Ces deux approches sont comparables dans le sens où elles obtiennent chacune toutes les solutions optimales dans le temps imparti. Néanmoins, Smet *et al.* se focalisent essentiellement sur la recherche de bonnes solutions et n'utilisent pas d'autre borne inférieure que celle donnée par M//. Autrement dit, cette approche n'est pas capable de prouver



(a) Data_137



(b) Data_100

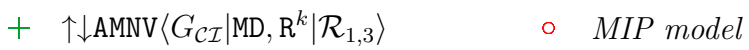


FIGURE 9.13 – Écart relatif entre $\underline{z}$ et $\overline{z}$, après 6 min.

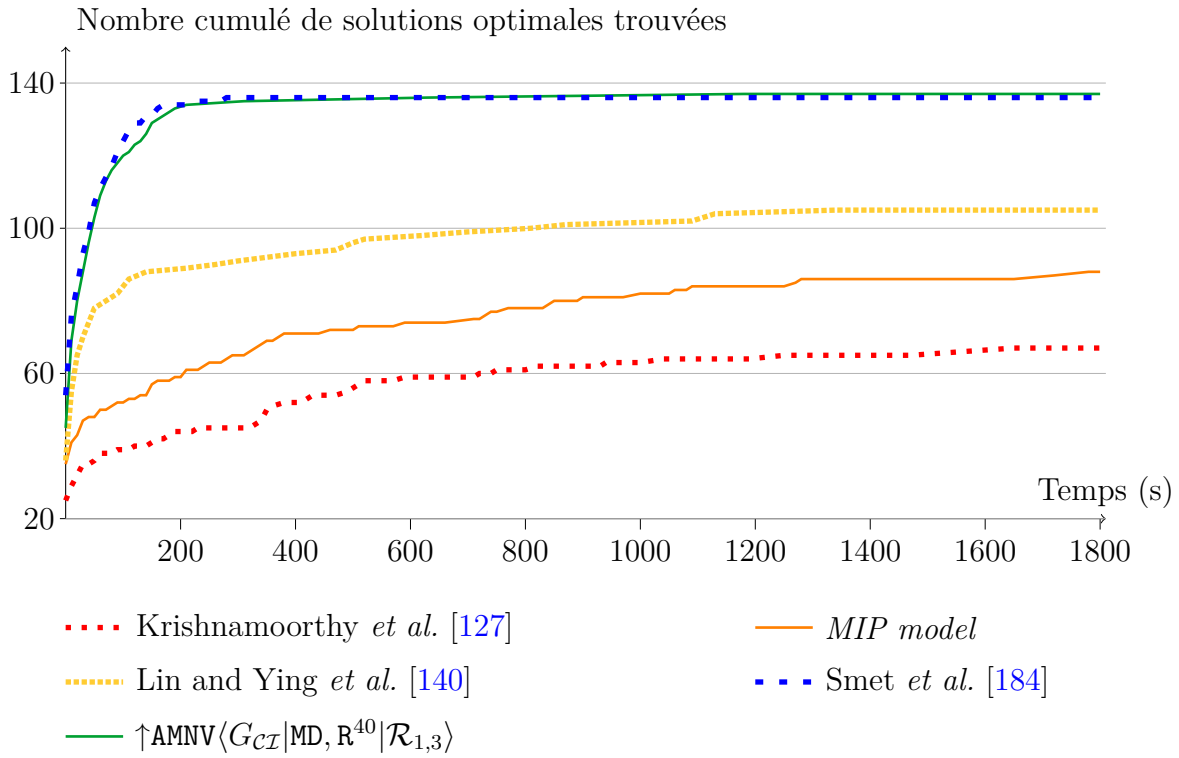


FIGURE 9.14 – Nombre cumulé de solutions optimales trouvées en fonction du temps, sur Data_137.

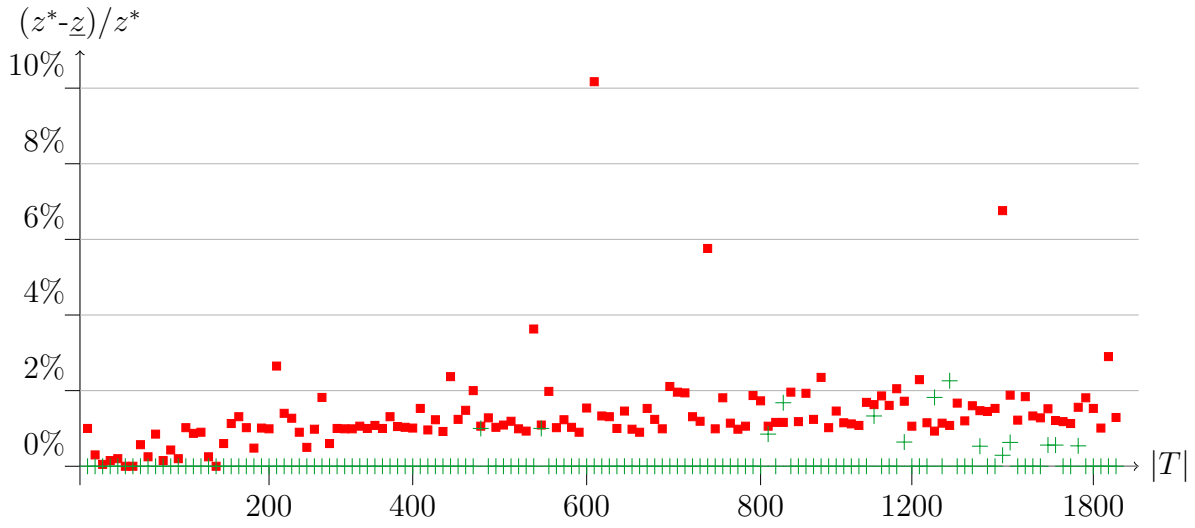
l'optimalité dans le cas général, contrairement à *CP model*.

Nous évaluons à présent la capacité de ↑AMNV⟨$G_{\mathcal{CI}}$|MD,R^k|$\mathcal{R}_{1,3}$⟩ à fournir rapidement une bonne borne inférieure. Pour cela, nous comparons la valeur et le temps de calcul de $\underline{z}_r$ avec ceux de la méthode proposée par Krishnamoorthy *et al.* [127]. Cette méthode étant fondée sur une relaxation Lagrangienne, nous notons $\underline{z}_{\mathcal{L}}$ la borne qu'elle permet d'obtenir. Dans le cadre d'une résolution complète, ↑AMNV⟨$G_{\mathcal{CI}}$|MD,R^{40}|$\mathcal{R}_{1,3}$⟩ donne les meilleurs résultats sur Data_137. Cependant, s'il ne s'agit que de calculer une borne inférieure, il peut être intéressant d'augmenter fortement k, de manière à réduire l'écart à l'optimum à moindre coût. Par conséquent, nous comparons $\underline{z}_{\mathcal{L}}$ avec $\underline{z}_r$ pour deux valeurs de k : 40 et 1 000. Pour faciliter la lecture des résultats, $\underline{z}_r$ est notée $\underline{z}_r^{40}$ quand $k = 40$ et $\underline{z}_r^{1000}$ quand k = 1 000. Bien entendu, nous n'utilisons pas M// dans notre modèle, puisque cela nous donnerait l'optimum sur toutes les instances.

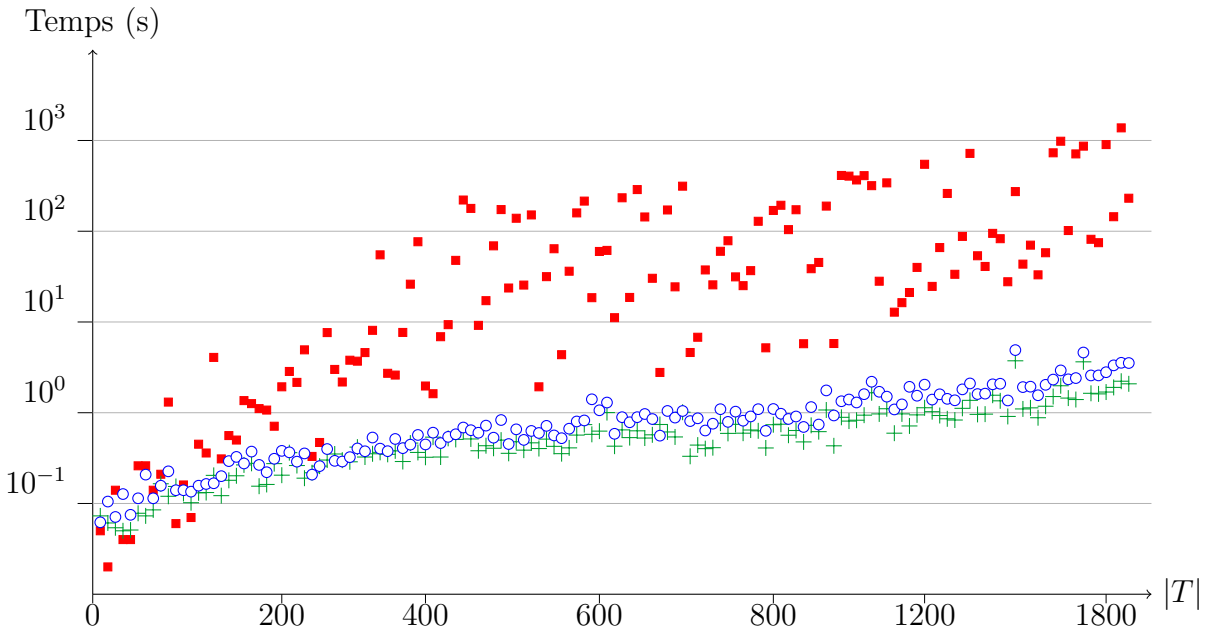
Comme cela est illustré par la Figure 9.15a, $\underline{z}_r^{40}$ et $\underline{z}_r^{1000}$ sont bien plus proches de z^* que $\underline{z}_{\mathcal{L}}$. Plus précisément, $\underline{z}_r^{1000}$ est toujours égale à z^* et $\underline{z}_r^{40}$ est en moyenne plus de dix fois plus proche de z^* que $\underline{z}_{\mathcal{L}}$. De plus, le plus grand écart relatif à l'optimum pour $\underline{z}_r^{40}$ est de 2.26% contre 10.17% pour $\underline{z}_{\mathcal{L}}$. Pour finir, comme cela est illustré par la Figure 9.15b, le temps de calcul de $\underline{z}_r^{40}$ et $\underline{z}_r^{1000}$ est bien plus faible que celui de $\underline{z}_{\mathcal{L}}$: le temps moyen de $\underline{z}_r^{40}$ tourne autour d'une demi-seconde contre plus d'une centaine de secondes pour $\underline{z}_{\mathcal{L}}$. Dans l'ensemble ↑AMNV⟨$G_{\mathcal{CI}}$|MD,R^k|$\mathcal{R}_{1,3}$⟩ est capable d'obtenir dès le nœud racine une très bonne borne inférieure, même sur de grandes instances.

9.5 Conclusions et perspectives

Nous avons présenté dans ce chapitre une approche performante pour le SMPTSP. Cette approche permet d'obtenir rapidement de bonnes bornes, même sur les instances de grande taille. Nous avons proposé la notation AMNV⟨G|F|$\mathcal{R}$⟩ pour décrire une famille de propagateur de la contrainte atMostNValue. Nous capitalisons sur les travaux de Bessière *et al.* [27] et proposons un nouveau



(a) Écart relatif entre $\underline{z}$ et z^* sur Data_137.



(b) Temps de calcul de $\underline{z}$ (en secondes), selon le modèle considéré.

$\underline{z}_{\mathcal{L}}$ $\underline{z}_r^{40}$ $\underline{z}_r^{1000}$

FIGURE 9.15 – Efficacité des méthodes de calcul de $\underline{z}$

propagateur $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{MD},\texttt{R}^k|\mathcal{R}_{1,3}\rangle$ pour filtrer la conjonction d'une contrainte `atMostNValue` avec des contraintes de différence. $\texttt{AMNV}\langle G_{\mathcal{CI}}|\texttt{MD},\texttt{R}^k|\mathcal{R}_{1,3}\rangle$ repose sur une structure de graphe plus appropriée ($G_{\mathcal{CI}}$), une nouvelle règle de filtrage ($\mathcal{R}_3$) et un mécanisme simple de diversification du filtrage ($\texttt{R}^k$). Pour finir, nous proposons une stratégie efficace de branchement reposant sur l'utilisation de contraintes globales réifiées.

De manière à prouver l'optimalité des instances tout en profitant au maximum du filtrage, nous avons opté pour une stratégie de branchement *bottom-up*. Si le but n'est pas de prouver l'optimalité mais de trouver rapidement une bonne solution, la stratégie *bottom-up* n'est pas adaptée, alors que la stratégie *top-down* est peu efficace. Il serait par conséquent intéressant d'améliorer cette stratégie ou de mettre en place une recherche par voisinage large de manière à obtenir rapidement des solutions proches de l'optimum. Par ailleurs, le mécanisme de branchement sur b_{reif} s'est révélé très efficace sur Data_137. Par conséquent, il pourrait être intéressant d'évaluer l'efficacité de ce genre de branchement sur d'autres problèmes. Ainsi, il serait envisageable d'étendre et de généraliser le branchement sur contraintes réifiées pour gérer des pans entiers de l'exploration d'un arbre de recherche. Les travaux présentés dans ce chapitre ont été publiés dans Artificial Intelligence [87].

Du point de vue de notre application industrielle, puisque l'approche que nous proposons permet de résoudre des instances à 2 000 tâches, il serait tout à fait possible de l'utiliser pour évaluer les besoins en intérimaires sur un horizon de plusieurs semaines. Par ailleurs, puisqu'il s'agit essentiellement d'évaluer les besoins en intérimaires, sans forcément détailler le planning de chaque employé, il serait envisageable de se contenter d'une bonne borne inférieure sur le nombre d'intérimaires. Dans ce cas, l'approche que nous proposons semble efficace et rapide. Il serait donc possible soit de travailler sur un horizon de plusieurs mois, soit de passer plus de temps sur la construction des horaires de travail des employés.

10

Conclusion

Sommaire

10.1 Travaux de thèse

Dans cette thèse, nous nous sommes tout d'abord intéressés au problème opérationnel de planification de personnel et d'affectation de tâches fixées dans le temps rencontré par Biotrial. Nous avons montré les limites d'une approche générique et proposé une méthode en deux phases donnant de bons résultats. Nous nous sommes ensuite intéressés à la gestion simultanée des problèmes de construction d'horaires et d'affectation de tâches via une méthode PPC puis avec une LNS. Le modèle PPC donne des résultats intéressants, mais globalement moins bons que ceux de la méthode en deux phases. L'analyse des résultats montre que cette méthode permet d'obtenir de bonnes solutions initiales, mais rencontre des difficultés à les améliorer. Contrairement au modèle PPC, la LNS bénéficie non seulement des mécanismes de filtrage de propagation, propres à la PPC, mais également des capacités de passage à l'échelle des métaheuristiques. Au final, cette méthode donne de meilleurs résultats que la méthode en deux phases, validant ainsi le principe d'une approche intégrée, mais non exacte. Nous avons conçu et développé un outil d'aide à la décision offrant un support adapté au problème de Biotrial. Cet outil permet notamment de trouver de bonnes solutions au problème de planification de personnel, grâce à une version légèrement modifiée de la LNS. De cette manière, les travaux réalisés durant cette thèse sont directement exploitables par Biotrial. Nous nous sommes également intéressés à un problème de la littérature, connexe à notre cas d'étude, mais présentant un cadre applicatif plus large. La méthode de résolution proposée s'est révélée efficace et compétitive avec les méthodes actuelles.

10.2 Perspectives

Plusieurs perspectives semblent intéressantes, tant sur le plan académique qu'industriel. Tout d'abord, comme signalé au Chapitre 8, la collecte des nouvelles données opérationnelles de Biotrial permettra d'évaluer les gains de temps lors de la planification du personnel ainsi que les gains financiers relatifs

aux embauches d'intérimaires. Cela nous permettra également de vérifier les performances des différentes méthodes développées durant cette thèse sur des données réelles, et de comparer nos solutions aux solution réelles. Il sera également envisageable de comparer les plannings automatiques avant et après modifications de manière à mesurer la qualité des plannings automatiques. Par ailleurs, une analyse plus fine des modifications apportées aux plannings automatiques nous permettrait sans doute d'améliorer la LNS utilisée par MAESTRO. En particulier, puisqu'il est envisagé d'affiner la fonction objectif en y intégrant, par exemple, la longueur et le positionnement des vacations, il serait utile de mettre en place une approche mieux à même d'arbitrer les conflits entre ces différents objectifs. Ces travaux n'ont pas été entrepris durant la thèse car l'obtention d'une solution complète est prioritaire par rapport à l'obtention d'une solution équitable, mais il n'en est pas forcément de même entre une solution équitable et une solution respectant les préférences des employés par exemple. D'ailleurs, la modélisation des préférences mériterait une étude approfondie car il s'agit d'un sujet complexe pour lequel une bonne modélisation peut fortement améliorer la satisfaction générale des employés. Dans un tel contexte, il pourrait être pertinent de développer des approches multi-objectifs [79] de manière à proposer aux décideurs une palette de quelques bonnes solutions, intéressantes selon différents aspects. Dans ce cas, il faudra alors convenir d'une représentation ergonomique des solutions pour que les décideurs soient en mesure de les comparer rapidement. En effet, l'analyse détaillée d'une solution automatique peut être assez fastidieuse. Pour finir sur les aspects métier, il serait également intéressant de concevoir de nouvelles méthodes de résolution, capables de gérer, par exemple, le positionnement des formations en doublon ou le lissage des horaires de travail sur le long terme.

Sur un plan plus académique et dans la lignée des travaux présentés, deux pistes principales retiennent notre attention : le développement de méthodes de résolution plus efficaces pour le SDPTSP||U,Δ, ainsi que l'étude de nouvelles variantes. En ce qui concerne la première piste, il est tout d'abord envisageable de chercher à développer une méthode permettant de calculer rapidement une bonne borne inférieure sur U. En effet, pour résoudre efficacement SDPTSP||U,Δ, l'idéal consiste en théorie à aborder l'objectif Δ après avoir atteint l'optimum sur U. De cette manière, cela évite de distinguer des solutions non optimales vis-à-vis de U, ce qui permet de réduire fortement l'espace de recherche. Néanmoins, en l'absence d'une bonne borne inférieure sur U, il semble plus efficace d'aborder simultanément U et Δ. Toujours dans l'optique de résoudre plus efficacement le SDPTSP||U,Δ, il pourrait être pertinent de développer une méthode de génération de colonnes, intégrée à une méthode de « branch and price ». Ce type d'approche s'est en effet révélée très efficace sur les problèmes de planification d'équipage dont la structure est proche de celle de notre cas d'étude. Puisqu'il s'agit d'une méthode exacte, cela permettrait de trouver de nouvelles solutions complètes ou, au contraire, de prouver davantage d'irréalisabilité.

En ce qui concerne la seconde piste, il semble naturel de poursuivre le travail initié avec le SMPTSP afin de concevoir une méthode de résolution pour le problème de dimensionnement d'effectif. Néanmoins, il faudrait alors déterminer d'une part les horaires de travail, mais également le profil de compétence des intérimaires, ce qui requiert une attention particulière, comme cela nous a été signalé par les décideurs de Biotrial. Puisque le profil de compétences des intérimaires est difficile à prévoir, un moyen de contourner cette difficulté peut consister à développer des approches capables de construire des plannings robustes vis-à-vis des compétences des intérimaires. Avec de tels plannings, il devient alors plus facile de réaffecter certaines tâches assignées à des intérimaires qui se révèlent insuffisamment formés sur certains actes médicaux. De manière plus générale, le développement d'une approche robuste semble incontournable dans le cadre de l'activité de Biotrial. En effet, l'activité de Biotrial étant entièrement fixée dans le temps, le moindre aléa peut avoir un effet catastrophique sur les plannings qui doivent alors être complètement revus. Autrement dit, le fait de considérer une activité fixée dans le temps réduit fortement la marge de manœuvre de Biotrial,

qui est alors très exposée aux aléas tels que que les absences imprévues, ou les études déplacées par exemple. Une approche robuste permettrait à Biotrial d'accroître cette marge de manœuvre. Bien qu'il soit essentiel de conserver un maximum de latitude en amont, il est également indispensable de gérer l'impact des aléas en aval. Cela revient en fait à développer une méthode de replanification qui permettra à Biotrial de faire face à un pic soudain d'activité, en adaptant les plannings tout en conservant au mieux les horaires de travail des employés et en automatisant la construction des horaires de recrutement d'intérimaires, même si des modifications manuelles restent possibles. Sur ce point les travaux de la littérature portant sur l'ordonnancement réactif [161, 185] pourraient donner de nombreuses idées, qu'il faudrait ensuite valider en pratique auprès des décideurs de Biotrial.

Lexique de contraintes

neq(x, y) stipule que les variables x et y sont différentes : $x \neq y$

eq(x, y) stipule que les variables x et y sont égales : $x = y$

geq(x, y) stipule que la variable x est supérieure ou égale à la variable y : $x \geq y$

leq(x, y) stipule que la variable x est inférieure ou égale à la variable y : $x \leq y$

scalar$(z, < x_1, .., x_n >, < c_1, .., c_n >)$ stipule que la variable z est égale à $\sum_{i=1}^{n} c_i x_i$

or$(c_1, .., c_n)$ stipule qu'au moins une des contraintes c est vérifiée : $c_1 \vee c_2 \vee ... \vee c_n$

ifOnlyIf(c_1, c_2) stipule que la contrainte c_1 est vérifiée ssi la contrainte c_2 est vérifiée : $c_1 \Leftrightarrow c_2$

card(s, c) stipule que la variable c est égale à card(s)

sumOverSet$(s, < c_1, .., c_n >, z)$ stipule que la variable z est égale à $\sum_{i \in s} c_i$

partition$(\mathcal{P}, < s_1, .., s_n >)$ stipule que les variables s_1 à s_n forment une partition de l'ensemble $\mathcal{P}$. Cela est équivalent à la conjonction de la contrainte **disjoint**$(< s_1, .., s_n >)$ avec la contrainte union$(\mathcal{P}, < s_1, .., s_n >)$

union$(u, < s_1, .., s_n >)$ stipule que la variable u est égale à $\bigcup_{i=1}^{n} s_i$

inter$(i, < s_1, s_2 >)$ stipule que la variable i est égale à $s_1 \cap s_2$

disjoint$(< s_1, .., s_n >)$ stipule que les variables s_i sont deux à deux disjointes :
$\forall (s_i, s_j), i \neq j, s_i \cap s_j = \emptyset$

maxOverSet$(s, < v_1, .., v_n >, z)$ stipule que la variable z est égale à $\max_{i \in s} \quad v_i$

minOverSet$(s, < v_1, .., v_n >, z)$ stipule que la variable z est égale à $\min_{i \in s} v_i$

count$(v, < x_1, .., x_n >, c)$ stipule que la variable c est égale à $\text{card}(\{x_i \in \{x_1, .., x_n\}, x_i = v\})$

channelingSetInt$(< s_1, .., s_n >, < x_1, .., x_m >)$ stipule que l'expression $t \in s_i$ équivaut à $x_t = i$.

allDifferent$(< x_1, .., x_n >)$ stipule que les variables x_1 à x_n prennent des valeurs différentes : $\forall i \neq j, x_i \neq x_j$

allDifferentBut$(v, < x_1, .., x_n >)$ stipule que les variables x_1 à x_n différentes de v prennent des valeurs différentes : $\forall (i, j)$ tels que $(i \neq j \wedge x_i \neq v \wedge x_j \neq v), x_i \neq x_j$

atMostNValue$(c, < x_1, .., x_n >)$ stipule qu'au plus c valeurs différentes apparaissent dans la collection de variables x_1 à x_n : $\text{card}(\{x_i, 1 \leq i \leq n\})$

B

Résultats des méthodes

Nous donnons ici les résultats des méthodes développées durant la thèse. La plupart des résultats sont donnés de manière agrégée selon les paramètre I_v et I_t, mais les meilleurs résultats de chaque approche sont également donnés à la Table B.14. Rappelons que les algorithmes sont implémentés en JAVA, que nous utilisons *Choco 3-1-1* [62] pour implémenter les modèles PPC, *Cplex 12.4* avec la configuration par défaut pour implémenter les modèles PLNE et que tous les tests sont réalisés avec un temps limite de 5 minutes sur un environnement identique : *Intel Core i3-540 (3.06 GHz & 8G RAM)*. Pour évaluer la qualité des résultats, nous prendrons en compte plusieurs indicateurs :

C : le nombre total d'instances pour lesquelles la méthode trouve une solution complète.
I : l'inéquité moyenne des solutions complètes (en minutes).
A : le pourcentage moyen de tâches affectées pour les solutions partielles uniquement.
T : le temps moyen d'obtention des meilleures solutions (en secondes).

Sommaire

B.1 Résultats de l'approche PPC

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	33/47	5/21	91/122
	I	95	151	216	122
	A	97,71	97,89	96,05	96,74
	T	161	152	96	136
200	C	59/60	42/55	9/35	110/150
	I	95	144	192	122
	A	99,00	98,78	97,71	98,00
	T	171	142	95	136
300	C	58/58	50/58	21/56	129/172
	I	89	131	261	133
	A	99,67	99,23	98,92	99,01
	T	201	178	99	159
400	C	59/59	54/59	34/56	147/174
	I	83	111	256	133
	A	99,75	99,63	99,07	99,19
	T	171	171	129	157
Total	C	229/231	179/219	69/168	**477/618**
	I	90	132	246	**128**
	A	98,37	98,54	97,66	**97,91**
	T	176	161	105	**147**

TABLE B.1 – CP-MW-LN-H_Δ

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	31/47	4/21	88/122
	I	89	101	62	92
	A	97,43	97,83	95,57	96,42
	T	137	117	95	117
200	C	59/60	44/55	8/35	111/150
	I	114	116	148	118
	A	99,00	98,72	97,37	97,70
	T	58	84	78	74
300	C	58/58	51/58	30/56	139/172
	I	122	121	115	120
	A	99,67	99,22	98,74	98,89
	T	30	36	30	32
400	C	59/59	52/59	32/56	143/174
	I	132	114	138	127
	A	99,75	99,66	98,96	99,14
	T	28	35	31	31
Total	C	229/231	178/219	74/168	**481/618**
	I	115	114	126	**116**
	A	98,19	98,50	97,28	**97,64**
	T	63	68	59	**63**

TABLE B.2 – CP-LW-LN-H_Δ

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	52/54	34/47	5/21	91/122
	I	102	168	301	138
	A	97,50	97,81	96,05	96,70
	T	187	134	99	140
200	C	56/60	44/55	13/35	113/150
	I	89	147	308	137
	A	99,25	98,88	97,50	97,93
	T	184	156	104	148
300	C	56/58	49/58	24/56	129/172
	I	117	172	284	169
	A	99,50	99,33	98,94	99,07
	T	185	145	93	141
400	C	53/59	47/59	29/56	129/174
	I	84	145	399	177
	A	99,71	99,67	99,03	99,29
	T	152	153	105	137
Total	C	217/231	174/219	71/168	**462/618**
	I	98	58	337	**157**
	A	98,83	98,69	97,62	**98,00**
	T	177	147	100	**141**

TABLE B.3 – CP-MW-BO-H_Δ

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	49/54	14/47	2/21	65/122
	I	91	127	190	102
	A	97,91	97,37	94,36	95,90
	T	155	119	98	124
200	C	55/60	25/55	5/35	85/150
	I	127	123	187	130
	A	99,40	98,80	96,53	97,52
	T	74	69	99	81
300	C	52/58	38/58	9/56	99/172
	I	142	131	194	142
	A	99,54	99,29	98,42	98,77
	T	40	65	66	57
400	C	52/59	38/59	9/56	99/174
	I	155	150	159	153
	A	99,72	99,63	98,88	99,17
	T	51	32	34	39
Total	C	208/231	115/219	25/168	**348/618**
	I	129	135	180	**135**
	A	99,00	98,50	96,95	**97,65**
	T	80	71	74	**75**

TABLE B.4 – CP-LW-BO-H_Δ

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	52/54	32/47	6/21	90/122
	I	114	185	372	156
	A	97,38	97,96	95,85	96,64
	T	148	120	108	125
200	C	55/60	44/55	7/35	106/150
	I	113	169	285	148
	A	99,20	98,72	97,16	97,64
	T	195	120	63	126
300	C	57/58	45/58	18/56	120/172
	I	146	187	214	172
	A	99,44	99,24	98,67	98,85
	T	159	131	68	119
400	C	53/59	47/59	19/56	119/174
	I	142	205	409	209
	A	99,68	99,60	98,86	99,11
	T	150	140	84	125
Total	C	217/231	168/219	50/168	**435/618**
	I	129	187	317	**173**
	A	98,74	98,69	97,49	**97,89**
	T	163	128	81	**124**

TABLE B.5 – CP-MW-ID-H_Δ

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	20/54	2/47	0/21	22/122
	I	140	319	-	156
	A	97,45	95,33	90,98	94,22
	T	92	62	57	70
200	C	16/60	1/55	0/35	17/150
	I	227	550	-	246
	A	98,85	96,50	92,40	95,63
	T	58	41	30	43
300	C	25/58	0/58	0/56	25/172
	I	296	-	-	296
	A	99,23	96,96	93,39	96,09
	T	33	49	48	43
400	C	20/59	1/59	0/56	21/174
	I	297	480	-	306
	A	99,34	97,69	94,33	96,83
	T	36	60	67	54
Total	C	81/231	4/219	0/168	**85/618**
	I	244	417	-	**252**
	A	98,71	96,63	92,77	**95,69**
	T	55	53	50	**53**

TABLE B.6 – CP-LW-ID-H_Δ

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	45/54	29/47	2/21	76/122
	I	129	245	288	178
	A	97,40	96,81	94,55	95,63
	T	178	138	120	145
200	C	54/60	32/55	3/35	89/150
	I	168	293	172	213
	A	99,42	98,18	96,41	97,15
	T	187	135	57	126
300	C	57/58	40/58	8/56	105/172
	I	226	306	493	277
	A	99,67	98,97	97,97	98,31
	T	178	142	58	126
400	C	52/59	43/59	12/56	107/174
	I	245	292	566	300
	A	99,66	99,51	98,39	98,79
	T	187	142	58	126
Total	C	208/231	144/219	25/168	**377/618**
	I	195	287	473	**248**
	A	98,55	98,14	96,73	**97,29**
	T	183	139	73	**132**

TABLE B.7 – CP-MW-BT-H_Δ

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	31/54	5/47	1/21	37/122
	I	26	34	46	28
	A	97,76	96,89	92,22	95,14
	T	55	73	75	68
200	C	41/60	5/55	0/35	46/150
	I	29	32	-	30
	A	99,26	98,05	94,92	96,82
	T	34	26	29	30
300	C	46/58	11/58	0/56	57/172
	I	29	34	-	30
	A	99,57	98,82	97,13	98,08
	T	25	25	32	27
400	C	41/59	12/59	1/56	54/174
	I	25	40	30	29
	A	99,68	99,03	97,60	98,46
	T	21	22	25	23
Total	C	159/231	33/219	2/172	**194/618**
	I	28	36	38	**29**
	A	98,88	98,15	95,47	**97,05**
	T	34	36	40	**37**

TABLE B.8 – CP-LW-BT-H_Δ

B.2 Résultats de l'approche M2P

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	42/47	11/21	106/122
	I	28	35	72	35
	A	97,50	97,78	96,71	97,05
	T	145	155	167	156
200	C	59/60	50/55	22/35	131/150
	I	34	35	42	36
	A	99,00	98,60	97,72	97,93
	T	166	138	156	154
300	C	58/58	53/58	42/56	153/172
	I	40	38	46	41
	A	99,67	99,19	98,76	98,94
	T	191	186	174	184
400	C	59/59	55/59	40/56	154/174
	I	47	44	51	47
	A	99,75	99,70	99,01	99,17
	T	236	202	196	211
Total	C	229/231	200/219	115/168	**544/618**
	I	38	38	49	**40**
	A	98,28	98,47	97,68	**97,90**
	T	184	170	173	**176**

TABLE B.9 – Solutions finales de M2P (5 min)

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	37/54	3/47	0/21	40/122
	I	1008	755	-	989
	A	97,78	96,54	91,30	94,50
	T	0,31	0,32	0,32	0,32
200	C	55/60	20/55	2/35	77/150
	I	1229	1004	934	1163
	A	99,00	98,40	94,70	96,34
	T	0,79	0,78	0,8	0,79
300	C	56/58	30/58	0/56	86/172
	I	1382	1114	-	1289
	A	99,42	98,88	97,42	97,97
	T	1,06	1,1	1,14	1,1
400	C	58/59	42/59	3/56	103/174
	I	1456	1207	989	1341
	A	99,75	99,18	98,11	98,41
	T	1,44	1,44	1,43	1,44
Total	C	206/231	95/219	5/168	**306/618**
	I	1295	1121	967	**1235**
	A	98,27	97,87	95,35	**96,94**
	T	0,9	0,91	0,92	**0,91**

TABLE B.10 – Solutions initiales de M2P

B.3 Résultats de l'approche LNS

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	54/54	46/47	19/21	119/122
	I	17	14	15	16
	A	97,67	98,07	97,17	97,43
	T	152	165	166	161
200	C	59/60	53/55	29/35	141/150
	I	27	24	28	26
	A	99,00	98,71	98,00	98,15
	T	115	131	142	129
300	C	58/58	54/58	47/56	159/172
	I	33	31	31	32
	A	99,67	99,33	98,90	99,10
	T	102	82	121	102
400	C	59/59	58/59	44/56	161/174
	I	34	31	39	34
	A	99,75	99,75	99,25	99,33
	T	179	146	160	162
Total	C	230/231	211/219	139/168	**580/618**
	I	28	26	31	**28**
	A	98,41	98,60	97,98	**98,14**
	T	137	131	147	**138**

TABLE B.11 – LNS avec H_Δ sur chaque solution.

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	54/54	46/47	20/21	120/122
	I	17	14	17	16
	A	97,67	98,07	97,10	97,38
	T	144	117	128	127
200	C	59/60	53/55	29/35	141/150
	I	27	24	26	26
	A	99,00	98,71	98,90	98,13
	T	73	175	162	162
300	C	58/58	54/58	47/56	159/172
	I	33	26	29	29
	A	99,67	99,33	98,90	99,10
	T	180	170	171	172
400	C	59/59	58/59	44/56	161/174
	I	34	31	38	34
	A	99,75	99,75	99,27	99,34
	T	278	289	195	209
Total	C	230/231	211/219	139/168	**581/618**
	I	28	24	30	**27**
	A	98,41	98,60	97,95	**98,12**
	T	158	154	155	**155**

TABLE B.12 – Meilleurs résultats de la LNS.

I_t	Crit.	I_v			
		600	800	1 000	Total
100	C	53/54	45/47	13/21	111/122
	I	13	14	74	20
	A	97,86	98,07	97,11	97,39
	T	158	119	122	138
200	C	59/60	52/55	27/35	138/150
	I	19	17	27	20
	A	99,00	98,81	97,98	98,17
	T	204	186	169	190
300	C	58/58	54/58	47/56	159/172
	I	28	25	42	31
	A	99,67	99,22	98,82	99,02
	T	219	206	223	216
400	C	59/59	58/59	43/56	160/174
	I	34	31	43	35
	A	99,75	99,75	99,25	99,33
	T	231	215	238	227
Total	C	229/231	209/219	130/168	**568/618**
	I	24	22	42	**28**
	A	98,46	98,59	97,90	**98,08**
	T	204	185	207	**198**

TABLE B.13 – LNS avec portée fixe.

B.4 Résultats détaillés

Le tableau ci-dessous donne les résultats détaillés des méthodes $PLNE_{||U}$ (10 heures) et $PPC_{||U,\Delta}$, LNS et M2P, au bout de 5 minutes. Le modèle $PPC_{||U,\Delta}$ est utilisé avec la stratégie combinée All-All, *i.e.* toutes les stratégies sont utilisées les unes à la suite des autres durant 37,5 secondes, soit un temps total de 5 minutes également.

Instance	$\underline{U}$	PLNE		PPC		LNS		M2P		Meilleur
		U	Δ	U	Δ	U	Δ	U	Δ	
Ta100-Ti600-SkC-i000	0	0	43	0	43	0	20	1	38	LNS
Ta100-Ti600-SkC-i001	0	0	230	0	51	0	18	0	28	LNS
Ta100-Ti600-SkC-i002	3	3	1278	3	63	3	56	4	41	LNS
Ta100-Ti600-SkC-i003	0	0	790	0	32	0	19	0	29	LNS
Ta100-Ti600-SkC-i004	0	0	67	0	34	0	14	0	27	LNS
Ta100-Ti600-SkC-i005	0	0	171	0	45	0	25	0	38	LNS
Ta100-Ti600-SkC-i006	0	0	37	0	53	0	15	0	29	LNS
Ta100-Ti600-SkC-i007	0	0	87	0	42	0	18	0	30	LNS
Ta100-Ti600-SkC-i008	0	0	211	0	29	0	9	0	22	LNS
Ta100-Ti600-SkC-i009	0	0	172	0	14	0	10	0	27	LNS
Ta100-Ti600-SkC-i010	0	0	115	0	38	0	19	0	24	LNS
Ta100-Ti600-SkC-i011	0	0	360	0	52	0	14	0	29	LNS
Ta100-Ti600-SkC-i012	0	0	52	0	42	0	22	0	29	LNS
Ta100-Ti600-SkC-i013	0	0	515	0	33	0	12	0	26	LNS
Ta100-Ti600-SkC-i014	0	0	115	0	78	0	12	0	42	LNS
Ta100-Ti600-SkC-i015	0	0	223	0	70	0	19	0	33	LNS
Ta100-Ti600-SkC-i016	0	0	41	0	35	0	10	0	33	LNS
Ta100-Ti600-SkC-i017	0	0	103	0	29	0	15	0	26	LNS
Ta100-Ti600-SkC-i018	0	0	187	0	43	0	18	0	35	LNS
Ta100-Ti600-SkC-i019	0	0	349	0	34	0	22	0	26	LNS
Ta100-Ti600-SkC-i020	0	0	243	0	62	0	28	0	36	LNS
Ta100-Ti600-SkC-i021	0	0	103	0	40	0	22	0	36	LNS
Ta100-Ti600-SkC-i022	1	1	1013	1	43	1	35	1	30	M2P
Ta100-Ti600-SkC-i023	2	2	1194	2	25	2	16	2	29	LNS
Ta100-Ti600-SkC-i024	0	0	36	0	35	0	8	0	17	LNS
Ta100-Ti600-SkC-i025	0	0	110	0	40	0	7	0	22	LNS
Ta100-Ti600-SkC-i026	0	0	247	0	37	0	28	0	22	M2P
Ta100-Ti600-SkC-i027	0	0	73	0	46	0	15	0	26	LNS
Ta100-Ti600-SkC-i028	0	0	132	0	48	0	11	0	26	LNS
Ta100-Ti600-SkC-i029	0	0	443	0	44	0	12	0	28	LNS
Ta100-Ti600-SkCR-i000	0	0	101	0	41	0	17	0	24	LNS
Ta100-Ti600-SkCR-i001	0	0	115	0	46	0	14	0	30	LNS
Ta100-Ti600-SkCR-i002	2	2	1274	2	74	2	36	2	39	LNS
Ta100-Ti600-SkCR-i003	0	0	288	0	37	0	17	0	24	LNS
Ta100-Ti600-SkCR-i004	0	0	128	0	32	0	22	0	30	LNS
Ta100-Ti600-SkCR-i005	0	0	106	0	36	0	15	0	26	LNS
Ta100-Ti600-SkCR-i006	0	0	161	0	38	0	13	0	24	LNS
Ta100-Ti600-SkCR-i007	0	0	314	0	42	0	19	0	33	LNS
Ta100-Ti600-SkCR-i008	0	0	260	0	42	0	16	0	34	LNS
Ta100-Ti600-SkCR-i009	0	0	295	0	35	0	19	0	26	LNS
Ta100-Ti600-SkCR-i010	0	0	175	0	39	0	13	0	30	LNS
Ta100-Ti600-SkCR-i011	0	0	250	0	43	0	26	0	29	LNS
Ta100-Ti600-SkCR-i012	0	0	124	0	44	0	17	0	19	LNS
Ta100-Ti600-SkCR-i013	2	2	1213	3	64	2	54	3	32	LNS
Ta100-Ti600-SkCR-i014	0	0	51	0	33	0	12	0	20	LNS
Ta100-Ti600-SkCR-i015	0	0	37	0	42	0	23	0	32	LNS
Ta100-Ti600-SkCR-i016	0	0	184	0	42	0	16	0	28	LNS
Ta100-Ti600-SkCR-i017	0	0	122	0	40	0	19	0	36	LNS
Ta100-Ti600-SkCR-i018	0	0	117	0	48	0	17	0	31	LNS
Ta100-Ti600-SkCR-i019	0	0	95	0	31	0	12	0	29	LNS
Ta100-Ti600-SkCR-i020	0	0	412	0	39	0	13	0	15	LNS
Ta100-Ti600-SkCR-i021	0	0	38	0	39	0	22	0	35	LNS
Ta100-Ti600-SkCR-i022	0	0	87	0	30	0	16	0	24	LNS
Ta100-Ti600-SkCR-i023	0	0	150	0	35	0	7	0	24	LNS
Ta100-Ti600-SkCR-i024	4	4	880	4	43	4	35	4	35	LNS-M2P
Ta100-Ti600-SkCR-i025	0	0	347	0	61	0	38	0	25	M2P
Ta100-Ti600-SkCR-i026	0	0	406	1	97	0	17	0	39	LNS
Ta100-Ti600-SkCR-i027	0	0	132	0	39	0	19	0	15	M2P
Ta100-Ti600-SkCR-i028	0	0	93	0	39	0	18	0	28	LNS
Ta100-Ti600-SkCR-i029	0	0	149	0	34	0	14	0	29	LNS
Ta100-Ti800-SkC-i000	0	0	315	0	114	0	21	0	29	LNS
Ta100-Ti800-SkC-i001	0	0	187	0	68	0	8	0	25	LNS
Ta100-Ti800-SkC-i002	0	0	737	0	53	0	4	0	20	LNS
Ta100-Ti800-SkC-i003	3	3	1143	5	44	3	98	4	25	LNS

Instance	$\underline{U}$	PLNE		PPC		LNS		M2P		Meilleur
		U	Δ	U	Δ	U	Δ	U	Δ	
Ta100-Ti800-SkC-i004	0	0	61	0	65	0	8	0	17	LNS
Ta100-Ti800-SkC-i005	0	0	422	1	101	0	10	0	28	LNS
Ta100-Ti800-SkC-i006	1	1	522	1	37	1	25	2	32	LNS
Ta100-Ti800-SkC-i007	1	1	981	3	105	1	61	2	42	LNS
Ta100-Ti800-SkC-i008	0	0	366	0	88	0	22	0	39	LNS
Ta100-Ti800-SkC-i009	0	0	30	0	46	0	14	0	24	LNS
Ta100-Ti800-SkC-i010	0	1	966	1	53	1	62	1	71	CP
Ta100-Ti800-SkC-i011	0	0	715	0	41	0	24	0	35	LNS
Ta100-Ti800-SkC-i012	0	0	440	0	66	0	13	0	22	LNS
Ta100-Ti800-SkC-i013	0	0	126	0	70	0	9	1	27	LNS
Ta100-Ti800-SkC-i014	0	0	159	2	77	0	7	0	20	LNS
Ta100-Ti800-SkC-i015	0	0	94	0	48	0	7	0	28	LNS
Ta100-Ti800-SkC-i016	0	0	354	0	55	0	13	0	27	LNS
Ta100-Ti800-SkC-i017	0	0	324	0	58	0	14	0	45	LNS
Ta100-Ti800-SkC-i018	0	0	620	0	45	0	10	0	24	LNS
Ta100-Ti800-SkC-i019	1	1	1052	2	52	1	29	2	28	LNS
Ta100-Ti800-SkC-i020	0	0	274	0	43	0	14	0	28	LNS
Ta100-Ti800-SkC-i021	0	0	62	0	25	0	10	0	25	LNS
Ta100-Ti800-SkC-i022	6	6	606	6	97	6	172	7	36	CP
Ta100-Ti800-SkC-i023	1	1	1101	1	42	1	67	1	27	M2P
Ta100-Ti800-SkC-i024	0	0	726	0	76	0	7	0	12	LNS
Ta100-Ti800-SkC-i025	0	0	532	0	72	0	10	0	43	LNS
Ta100-Ti800-SkC-i026	1	1	821	1	58	1	24	2	23	LNS
Ta100-Ti800-SkC-i027	0	0	250	0	48	0	17	0	29	LNS
Ta100-Ti800-SkC-i028	1	1	785	1	56	1	35	1	37	LNS
Ta100-Ti800-SkC-i029	0	0	273	1	53	0	26	1	35	LNS
Ta100-Ti800-SkCR-i000	0	0	391	0	40	0	10	0	24	LNS
Ta100-Ti800-SkCR-i001	0	0	824	0	82	0	15	0	41	LNS
Ta100-Ti800-SkCR-i002	4	4	861	5	48	4	32	4	27	M2P
Ta100-Ti800-SkCR-i003	0	0	322	0	56	0	14	1	30	LNS
Ta100-Ti800-SkCR-i004	0	0	691	0	46	0	16	0	27	LNS
Ta100-Ti800-SkCR-i005	0	0	101	0	63	0	9	0	22	LNS
Ta100-Ti800-SkCR-i006	0	0	781	2	124	0	13	0	27	LNS
Ta100-Ti800-SkCR-i007	0	0	469	0	56	0	25	0	23	M2P
Ta100-Ti800-SkCR-i008	0	0	100	0	48	0	7	1	31	LNS
Ta100-Ti800-SkCR-i009	0	0	666	0	121	0	37	0	283	LNS
Ta100-Ti800-SkCR-i010	0	0	236	0	104	0	10	0	40	LNS
Ta100-Ti800-SkCR-i011	0	0	382	0	32	0	16	0	17	LNS
Ta100-Ti800-SkCR-i012	0	0	784	0	68	0	11	0	14	LNS
Ta100-Ti800-SkCR-i013	0	0	177	0	61	0	13	0	29	LNS
Ta100-Ti800-SkCR-i014	2	2	865	2	153	2	56	2	36	M2P
Ta100-Ti800-SkCR-i015	2	2	1105	2	60	2	31	2	32	LNS
Ta100-Ti800-SkCR-i016	2	2	598	4	69	2	49	3	25	LNS
Ta100-Ti800-SkCR-i017	0	0	554	0	57	0	15	0	18	LNS
Ta100-Ti800-SkCR-i018	0	0	230	0	55	0	11	0	24	LNS
Ta100-Ti800-SkCR-i019	0	0	321	0	87	0	15	0	25	LNS
Ta100-Ti800-SkCR-i020	0	0	423	0	35	0	13	0	34	LNS
Ta100-Ti800-SkCR-i021	0	0	222	0	44	0	11	0	23	LNS
Ta100-Ti800-SkCR-i022	0	0	271	0	98	0	15	0	27	LNS
Ta100-Ti800-SkCR-i023	0	0	903	1	150	0	13	0	85	LNS
Ta100-Ti800-SkCR-i024	0	0	44	0	78	0	12	0	29	LNS
Ta100-Ti800-SkCR-i025	0	0	239	0	58	0	23	0	32	LNS
Ta100-Ti800-SkCR-i026	0	0	607	0	74	0	23	0	35	LNS
Ta100-Ti800-SkCR-i027	1	1	1421	1	81	1	52	3	32	LNS
Ta100-Ti800-SkCR-i028	0	0	118	0	71	0	34	0	40	LNS
Ta100-Ti800-SkCR-i029	0	0	85	0	56	0	5	0	29	LNS
Ta100-Ti1000-SkC-i000	2	2	900	2	67	2	18	2	18	LNS-M2P
Ta100-Ti1000-SkC-i001	3	3	742	5	122	3	45	3	23	M2P
Ta100-Ti1000-SkC-i002	2	2	1164	4	69	2	252	3	34	LNS
Ta100-Ti1000-SkC-i003	2	2	849	4	58	2	29	2	26	M2P
Ta100-Ti1000-SkC-i004	0	0	1282	0	55	0	9	1	31	LNS
Ta100-Ti1000-SkC-i005	0	0	215	1	39	0	8	0	16	LNS
Ta100-Ti1000-SkC-i006	0	0	885	0	148	0	33	0	40	LNS
Ta100-Ti1000-SkC-i007	3	3	1127	4	205	3	79	3	44	M2P
Ta100-Ti1000-SkC-i008	0	0	685	1	89	0	8	2	133	LNS
Ta100-Ti1000-SkC-i009	5	5	932	7	152	5	78	6	19	LNS
Ta100-Ti1000-SkC-i010	2	2	1118	3	79	2	24	2	26	LNS
Ta100-Ti1000-SkC-i011	1	1	1209	2	51	1	116	2	38	LNS
Ta100-Ti1000-SkC-i012	0	0	388	0	104	0	9	2	33	LNS
Ta100-Ti1000-SkC-i013	6	6	910	7	133	6	33	8	45	LNS
Ta100-Ti1000-SkC-i014	2	2	1464	2	94	2	109	3	36	CP
Ta100-Ti1000-SkC-i015	2	2	814	5	79	2	72	3	27	LNS
Ta100-Ti1000-SkC-i016	1	1	1091	1	40	1	37	1	37	LNS-M2P
Ta100-Ti1000-SkC-i017	1	1	728	3	78	1	229	1	26	M2P

Instance	$\underline{U}$	PLNE		PPC		LNS		M2P		Meilleur
		U	Δ	U	Δ	U	Δ	U	Δ	
Ta100-Ti1000-SkC-i018	0	1	1055	2	106	1	40	2	25	LNS
Ta100-Ti1000-SkC-i019	3	3	1413	3	127	3	97	3	40	M2P
Ta100-Ti1000-SkC-i020	5	6	1326	8	47	6	223	6	25	M2P
Ta100-Ti1000-SkC-i021	2	2	1253	3	109	2	20	3	19	LNS
Ta100-Ti1000-SkC-i022	0	0	783	0	58	0	13	0	17	LNS
Ta100-Ti1000-SkC-i023	3	3	857	4	117	4	100	5	24	MIP
Ta100-Ti1000-SkC-i024	1	1	1158	3	177	1	84	1	27	M2P
Ta100-Ti1000-SkC-i025	0	0	452	0	57	0	12	0	23	LNS
Ta100-Ti1000-SkC-i026	1	1	1244	1	22	1	78	2	30	CP
Ta100-Ti1000-SkC-i027	0	0	583	0	185	0	23	1	72	LNS
Ta100-Ti1000-SkC-i028	1	1	992	3	61	1	108	3	25	LNS
Ta100-Ti1000-SkC-i029	0	0	359	1	32	0	29	0	49	LNS
Ta100-Ti1000-SkCR-i000	4	5	1076	7	43	5	173	5	34	M2P
Ta100-Ti1000-SkCR-i001	1	1	828	3	86	1	340	2	30	LNS
Ta100-Ti1000-SkCR-i002	2	2	1133	2	112	2	25	2	20	M2P
Ta100-Ti1000-SkCR-i003	0	0	669	3	91	0	21	0	72	LNS
Ta100-Ti1000-SkCR-i004	8	8	1028	10	86	8	188	8	290	LNS
Ta100-Ti1000-SkCR-i005	0	0	236	1	126	0	11	0	30	LNS
Ta100-Ti1000-SkCR-i006	2	2	1513	2	60	2	597	3	37	CP
Ta100-Ti1000-SkCR-i007	8	8	1168	12	114	8	117	8	12	M2P
Ta100-Ti1000-SkCR-i008	3	3	1293	6	116	3	87	3	54	M2P
Ta100-Ti1000-SkCR-i009	0	0	993	3	75	1	336	0	469	M2P
Ta100-Ti1000-SkCR-i010	1	1	1143	4	100	1	90	2	31	LNS
Ta100-Ti1000-SkCR-i011	0	0	300	3	140	0	33	2	76	LNS
Ta100-Ti1000-SkCR-i012	3	4	724	3	97	3	33	6	26	LNS
Ta100-Ti1000-SkCR-i013	2	2	1736	3	90	2	123	3	28	LNS
Ta100-Ti1000-SkCR-i014	5	5	573	10	76	5	161	8	24	LNS
Ta100-Ti1000-SkCR-i015	0	0	279	0	88	0	6	1	16	LNS
Ta100-Ti1000-SkCR-i016	6	6	1493	7	101	6	181	6	100	M2P
Ta100-Ti1000-SkCR-i017	0	0	319	0	53	0	17	0	36	LNS
Ta100-Ti1000-SkCR-i018	1	1	722	2	46	1	121	3	20	LNS
Ta100-Ti1000-SkCR-i019	0	0	677	3	202	0	9	1	100	LNS
Ta100-Ti1000-SkCR-i020	0	0	626	2	137	0	13	2	37	LNS
Ta100-Ti1000-SkCR-i021	1	1	1458	2	130	1	124	1	23	M2P
Ta100-Ti1000-SkCR-i022	2	3	1062	5	106	3	67	3	26	M2P
Ta100-Ti1000-SkCR-i023	0	0	189	1	36	0	19	0	20	LNS
Ta100-Ti1000-SkCR-i024	0	0	208	3	91	0	8	1	31	LNS
Ta100-Ti1000-SkCR-i025	1	1	745	3	80	1	238	5	20	LNS
Ta100-Ti1000-SkCR-i026	7	7	1220	9	71	7	92	8	41	LNS
Ta100-Ti1000-SkCR-i027	1	1	1155	3	101	1	155	2	29	LNS
Ta100-Ti1000-SkCR-i028	0	0	564	0	90	0	6	0	25	LNS
Ta100-Ti1000-SkCR-i029	4	4	942	9	33	4	27	6	30	LNS
Ta200-Ti600-SkC-i000	0	0	171	0	24	0	28	0	30	CP
Ta200-Ti600-SkC-i001	0	0	192	0	46	0	32	0	36	LNS
Ta200-Ti600-SkC-i002	0	0	685	0	25	0	18	0	45	LNS
Ta200-Ti600-SkC-i003	0	2	1151	2	42	2	40	2	51	LNS
Ta200-Ti600-SkC-i004	0	0	37	0	42	0	26	0	40	LNS
Ta200-Ti600-SkC-i005	0	0	50	0	22	0	17	0	31	LNS
Ta200-Ti600-SkC-i006	0	0	121	0	33	0	40	0	32	M2P
Ta200-Ti600-SkC-i007	0	0	173	0	29	0	16	0	50	LNS
Ta200-Ti600-SkC-i008	0	0	73	0	38	0	27	0	41	LNS
Ta200-Ti600-SkC-i009	0	0	94	0	35	0	22	0	38	LNS
Ta200-Ti600-SkC-i010	0	0	82	0	33	0	26	0	26	LNS-M2P
Ta200-Ti600-SkC-i011	0	0	221	0	36	0	31	0	32	LNS
Ta200-Ti600-SkC-i012	0	0	366	0	40	0	42	0	28	M2P
Ta200-Ti600-SkC-i013	0	0	190	0	41	0	28	0	41	LNS
Ta200-Ti600-SkC-i014	0	0	49	0	32	0	21	0	29	LNS
Ta200-Ti600-SkC-i015	0	0	368	0	42	0	34	0	42	LNS
Ta200-Ti600-SkC-i016	0	0	160	0	16	0	31	0	24	CP
Ta200-Ti600-SkC-i017	0	0	141	0	19	0	16	0	20	LNS
Ta200-Ti600-SkC-i018	0	0	59	0	33	0	25	0	34	LNS
Ta200-Ti600-SkC-i019	0	0	109	0	44	0	25	0	45	LNS
Ta200-Ti600-SkC-i020	0	0	54	0	30	0	24	0	24	LNS-M2P
Ta200-Ti600-SkC-i021	0	0	171	0	25	0	30	0	20	M2P
Ta200-Ti600-SkC-i022	0	0	121	0	37	0	22	0	25	LNS
Ta200-Ti600-SkC-i023	0	0	68	0	33	0	20	0	29	LNS
Ta200-Ti600-SkC-i024	0	0	68	0	24	0	27	0	25	CP
Ta200-Ti600-SkC-i025	0	0	336	0	50	0	17	0	32	LNS
Ta200-Ti600-SkC-i026	0	0	535	0	31	0	23	0	37	LNS
Ta200-Ti600-SkC-i027	0	0	91	0	35	0	26	0	34	LNS
Ta200-Ti600-SkC-i028	0	0	73	0	29	0	20	0	29	LNS
Ta200-Ti600-SkC-i029	0	0	152	0	31	0	30	0	40	LNS
Ta200-Ti600-SkCR-i000	0	0	52	0	37	0	26	0	29	LNS
Ta200-Ti600-SkCR-i001	0	0	61	0	39	0	33	0	35	LNS

Instance	U	PLNE		PPC		LNS		M2P		Meilleur
		U	Δ	U	Δ	U	Δ	U	Δ	
Ta200-Ti600-SkCR-i002	0	0	100	0	42	0	20	0	22	LNS
Ta200-Ti600-SkCR-i003	0	0	99	0	29	0	30	0	29	CP-M2P
Ta200-Ti600-SkCR-i004	0	0	45	0	33	0	30	0	51	LNS
Ta200-Ti600-SkCR-i005	0	0	169	0	36	0	31	0	48	LNS
Ta200-Ti600-SkCR-i006	0	0	165	0	35	0	36	0	36	CP
Ta200-Ti600-SkCR-i007	0	0	106	0	53	0	48	0	29	M2P
Ta200-Ti600-SkCR-i008	0	0	41	0	34	0	31	0	43	LNS
Ta200-Ti600-SkCR-i009	0	0	26	0	38	0	25	0	40	LNS
Ta200-Ti600-SkCR-i010	0	0	378	0	24	0	27	0	33	CP
Ta200-Ti600-SkCR-i011	0	0	36	0	39	0	16	0	35	LNS
Ta200-Ti600-SkCR-i012	0	0	20	0	34	0	28	0	34	MIP
Ta200-Ti600-SkCR-i013	0	0	137	0	41	0	24	0	30	LNS
Ta200-Ti600-SkCR-i014	0	0	80	0	22	0	18	0	27	LNS
Ta200-Ti600-SkCR-i015	0	0	53	0	44	0	33	0	54	LNS
Ta200-Ti600-SkCR-i016	0	0	67	0	28	0	26	0	33	LNS
Ta200-Ti600-SkCR-i017	0	0	601	0	53	0	26	0	31	LNS
Ta200-Ti600-SkCR-i018	0	0	196	0	28	0	22	0	35	LNS
Ta200-Ti600-SkCR-i019	0	0	62	0	34	0	23	0	38	LNS
Ta200-Ti600-SkCR-i020	0	0	81	0	36	0	23	0	39	LNS
Ta200-Ti600-SkCR-i021	0	0	450	0	29	0	28	0	37	LNS
Ta200-Ti600-SkCR-i022	0	0	52	0	38	0	24	0	33	LNS
Ta200-Ti600-SkCR-i023	0	0	170	0	40	0	31	0	25	M2P
Ta200-Ti600-SkCR-i024	0	0	111	0	46	0	29	0	33	LNS
Ta200-Ti600-SkCR-i025	0	0	139	0	32	0	32	0	33	CP-LNS
Ta200-Ti600-SkCR-i026	0	0	392	0	45	0	30	0	37	LNS
Ta200-Ti600-SkCR-i027	0	0	510	0	33	0	28	0	26	M2P
Ta200-Ti600-SkCR-i028	0	0	220	0	42	0	23	0	35	LNS
Ta200-Ti600-SkCR-i029	0	0	106	0	49	0	29	0	39	LNS
Ta200-Ti800-SkC-i000	0	0	118	0	43	0	19	0	22	LNS
Ta200-Ti800-SkC-i001	0	0	240	0	49	0	22	0	31	LNS
Ta200-Ti800-SkC-i002	0	0	140	0	28	0	21	0	34	LNS
Ta200-Ti800-SkC-i003	0	0	833	1	65	0	21	2	64	LNS
Ta200-Ti800-SkC-i004	0	0	124	0	49	0	37	0	37	LNS-M2P
Ta200-Ti800-SkC-i005	0	0	436	0	33	0	23	0	31	LNS
Ta200-Ti800-SkC-i006	0	0	205	0	36	0	30	0	31	LNS
Ta200-Ti800-SkC-i007	0	0	574	0	50	0	34	0	37	LNS
Ta200-Ti800-SkC-i008	2	4	1031	4	38	3	40	3	37	M2P
Ta200-Ti800-SkC-i009	0	0	96	0	41	0	15	0	26	LNS
Ta200-Ti800-SkC-i010	0	0	116	0	34	0	23	0	25	LNS
Ta200-Ti800-SkC-i011	0	0	542	0	48	0	25	0	27	LNS
Ta200-Ti800-SkC-i012	0	2	1409	0	41	0	23	0	38	LNS
Ta200-Ti800-SkC-i013	0	1	1158	0	50	0	30	0	32	LNS
Ta200-Ti800-SkC-i014	0	2	1109	1	52	0	28	1	40	LNS
Ta200-Ti800-SkC-i015	0	0	82	0	49	0	15	0	33	LNS
Ta200-Ti800-SkC-i016	0	0	97	0	32	0	24	0	34	LNS
Ta200-Ti800-SkC-i017	0	0	140	0	48	0	23	0	36	LNS
Ta200-Ti800-SkC-i018	0	0	352	0	40	0	12	0	20	LNS
Ta200-Ti800-SkC-i019	2	3	1255	5	65	3	100	7	58	LNS
Ta200-Ti800-SkC-i020	0	3	1264	1	61	1	36	2	78	LNS
Ta200-Ti800-SkC-i021	0	0	67	0	31	0	20	0	36	LNS
Ta200-Ti800-SkC-i022	0	0	478	0	49	0	24	0	47	LNS
Ta200-Ti800-SkC-i023	0	0	586	0	42	0	20	0	29	LNS
Ta200-Ti800-SkC-i024	0	0	378	0	46	0	30	0	35	LNS
Ta200-Ti800-SkC-i025	0	0	159	0	50	0	35	0	59	LNS
Ta200-Ti800-SkC-i026	0	1	1476	2	48	0	26	0	35	LNS
Ta200-Ti800-SkC-i027	0	0	70	0	46	0	27	0	42	LNS
Ta200-Ti800-SkC-i028	0	0	335	0	49	0	18	0	26	LNS
Ta200-Ti800-SkC-i029	0	0	118	0	89	0	33	0	34	LNS
Ta200-Ti800-SkCR-i000	0	0	77	0	39	0	31	0	27	M2P
Ta200-Ti800-SkCR-i001	0	0	223	0	67	0	31	0	21	M2P
Ta200-Ti800-SkCR-i002	0	2	1371	0	36	0	22	0	32	LNS
Ta200-Ti800-SkCR-i003	0	0	297	0	46	0	27	0	38	LNS
Ta200-Ti800-SkCR-i004	0	0	50	0	49	0	23	0	29	LNS
Ta200-Ti800-SkCR-i005	0	1	1103	0	49	0	31	1	56	LNS
Ta200-Ti800-SkCR-i006	0	0	602	0	75	0	24	0	33	LNS
Ta200-Ti800-SkCR-i007	0	0	299	0	39	0	31	0	30	M2P
Ta200-Ti800-SkCR-i008	0	0	374	0	29	0	15	0	51	LNS
Ta200-Ti800-SkCR-i009	0	0	614	0	37	0	30	0	33	LNS
Ta200-Ti800-SkCR-i010	0	0	536	0	59	0	17	0	28	LNS
Ta200-Ti800-SkCR-i011	0	0	309	0	49	0	36	0	47	LNS
Ta200-Ti800-SkCR-i012	0	0	139	0	49	0	10	0	40	LNS
Ta200-Ti800-SkCR-i013	0	0	121	0	32	0	21	0	30	LNS
Ta200-Ti800-SkCR-i014	0	0	81	0	44	0	19	0	32	LNS
Ta200-Ti800-SkCR-i015	2	3	1263	4	41	3	38	4	33	LNS

Instance	U	PLNE		PPC		LNS		M2P		Meilleur
		U	Δ	U	Δ	U	Δ	U	Δ	
Ta200-Ti800-SkCR-i016	0	0	207	0	40	0	28	0	27	M2P
Ta200-Ti800-SkCR-i017	2	4	1496	5	51	4	63	4	54	M2P
Ta200-Ti800-SkCR-i018	0	1	1517	1	36	0	24	0	65	LNS
Ta200-Ti800-SkCR-i019	0	1	816	0	50	0	20	0	47	LNS
Ta200-Ti800-SkCR-i020	0	0	509	0	61	0	19	0	29	LNS
Ta200-Ti800-SkCR-i021	0	0	55	0	52	0	26	0	25	M2P
Ta200-Ti800-SkCR-i022	0	0	73	1	37	0	22	0	29	LNS
Ta200-Ti800-SkCR-i023	0	0	567	0	38	0	18	0	40	LNS
Ta200-Ti800-SkCR-i024	0	0	100	0	40	0	33	0	46	LNS
Ta200-Ti800-SkCR-i025	0	0	78	0	43	0	29	0	40	LNS
Ta200-Ti800-SkCR-i026	0	0	109	0	33	0	16	0	50	LNS
Ta200-Ti800-SkCR-i027	2	3	1337	3	61	2	64	2	58	M2P
Ta200-Ti800-SkCR-i028	0	3	952	3	62	2	28	2	46	LNS
Ta200-Ti800-SkCR-i029	0	0	47	0	41	0	17	0	34	LNS
Ta200-Ti1000-SkC-i000	0	0	670	4	68	0	16	0	29	LNS
Ta200-Ti1000-SkC-i001	0	1	1441	0	59	0	28	0	38	LNS
Ta200-Ti1000-SkC-i002	2	4	1266	4	84	4	29	4	54	LNS
Ta200-Ti1000-SkC-i003	4	5	943	10	91	5	248	6	54	LNS
Ta200-Ti1000-SkC-i004	1	4	1162	3	61	3	76	4	59	CP
Ta200-Ti1000-SkC-i005	2	2	1361	3	77	2	53	4	46	LNS
Ta200-Ti1000-SkC-i006	2	3	1929	3	57	2	167	2	77	M2P
Ta200-Ti1000-SkC-i007	0	0	199	1	36	0	19	0	50	LNS
Ta200-Ti1000-SkC-i008	0	2	1246	0	54	0	46	4	53	LNS
Ta200-Ti1000-SkC-i009	0	0	209	0	63	0	18	0	39	LNS
Ta200-Ti1000-SkC-i010	0	1	1284	0	65	0	28	1	34	LNS
Ta200-Ti1000-SkC-i011	0	0	215	0	49	0	15	0	38	LNS
Ta200-Ti1000-SkC-i012	0	0	517	0	79	0	25	1	43	LNS
Ta200-Ti1000-SkC-i013	3	4	1327	5	54	4	42	5	44	LNS
Ta200-Ti1000-SkC-i014	0	0	456	3	97	0	16	2	35	LNS
Ta200-Ti1000-SkC-i015	2	6	1343	6	59	5	49	5	49	LNS-M2P
Ta200-Ti1000-SkC-i016	1	3	838	5	54	3	94	5	72	LNS
Ta200-Ti1000-SkC-i017	0	1	1295	0	73	0	29	0	71	LNS
Ta200-Ti1000-SkC-i018	2	5	1583	5	43	5	78	6	35	CP
Ta200-Ti1000-SkC-i019	0	2	1148	2	58	2	67	2	48	M2P
Ta200-Ti1000-SkC-i020	3	7	833	6	72	6	67	6	73	LNS
Ta200-Ti1000-SkC-i021	2	6	1012	7	98	6	106	6	77	M2P
Ta200-Ti1000-SkC-i022	1	3	1210	6	159	3	89	5	55	LNS
Ta200-Ti1000-SkC-i023	2	10	1045	11	99	8	151	12	41	LNS
Ta200-Ti1000-SkC-i024	0	1	1431	2	99	0	28	0	41	LNS
Ta200-Ti1000-SkC-i025	0	0	447	0	81	0	18	0	48	LNS
Ta200-Ti1000-SkC-i026	0	2	1087	3	62	2	48	3	60	LNS
Ta200-Ti1000-SkC-i027	1	4	1523	3	76	3	105	5	119	CP
Ta200-Ti1000-SkC-i028	0	0	161	0	55	0	26	0	37	LNS
Ta200-Ti1000-SkC-i029	7	12	744	11	55	11	58	11	38	M2P
Ta200-Ti1000-SkCR-i000	0	0	471	0	66	0	24	0	32	LNS
Ta200-Ti1000-SkCR-i001	0	0	744	3	171	0	45	0	85	LNS
Ta200-Ti1000-SkCR-i002	0	1	1561	0	69	0	38	0	40	LNS
Ta200-Ti1000-SkCR-i003	0	0	759	0	71	0	27	1	66	LNS
Ta200-Ti1000-SkCR-i004	0	1	1267	2	56	0	13	1	75	LNS
Ta200-Ti1000-SkCR-i005	1	5	1366	5	49	2	103	3	47	LNS
Ta200-Ti1000-SkCR-i006	1	3	1391	6	85	3	114	3	161	LNS
Ta200-Ti1000-SkCR-i007	0	0	39	0	66	0	32	0	37	LNS
Ta200-Ti1000-SkCR-i008	0	0	179	0	67	0	35	0	40	LNS
Ta200-Ti1000-SkCR-i009	3	8	1446	9	76	7	130	8	41	LNS
Ta200-Ti1000-SkCR-i010	0	2	1566	2	53	1	338	2	69	LNS
Ta200-Ti1000-SkCR-i011	7	7	1298	11	72	7	109	8	94	LNS
Ta200-Ti1000-SkCR-i012	0	4	1292	6	58	1	198	6	41	LNS
Ta200-Ti1000-SkCR-i013	3	3	1207	4	71	3	144	4	40	LNS
Ta200-Ti1000-SkCR-i014	0	0	637	0	76	0	62	0	36	M2P
Ta200-Ti1000-SkCR-i015	0	0	148	2	79	0	39	0	61	LNS
Ta200-Ti1000-SkCR-i016	1	3	1388	4	97	3	44	3	77	LNS
Ta200-Ti1000-SkCR-i017	0	0	394	3	63	0	24	1	72	LNS
Ta200-Ti1000-SkCR-i018	0	3	1079	10	83	3	404	4	47	LNS
Ta200-Ti1000-SkCR-i019	0	0	485	1	91	0	23	0	28	LNS
Ta200-Ti1000-SkCR-i020	2	2	1134	2	56	2	59	2	69	CP
Ta200-Ti1000-SkCR-i021	0	2	1380	3	48	0	24	0	42	LNS
Ta200-Ti1000-SkCR-i022	4	9	1078	12	68	6	230	9	53	LNS
Ta200-Ti1000-SkCR-i023	0	1	1559	0	54	0	33	0	42	LNS
Ta200-Ti1000-SkCR-i024	0	0	232	0	56	0	35	0	34	M2P
Ta200-Ti1000-SkCR-i025	0	5	1289	8	69	2	438	6	74	LNS
Ta200-Ti1000-SkCR-i026	0	1	1467	0	58	0	20	0	22	LNS
Ta200-Ti1000-SkCR-i027	5	7	866	10	68	7	101	9	28	LNS
Ta200-Ti1000-SkCR-i028	1	4	1587	5	98	3	188	4	62	LNS
Ta200-Ti1000-SkCR-i029	0	2	1314	0	55	0	27	0	35	LNS

Instance	$\underline{U}$	PLNE		PPC		LNS		M2P		Meilleur
		U	Δ	U	Δ	U	Δ	U	Δ	
Ta300-Ti600-SkC-i000	0	0	279	0	22	0	30	0	40	CP
Ta300-Ti600-SkC-i001	0	0	82	0	23	0	26	0	28	CP
Ta300-Ti600-SkC-i002	0	0	411	0	35	0	60	0	40	CP
Ta300-Ti600-SkC-i003	0	0	51	0	18	0	25	0	40	CP
Ta300-Ti600-SkC-i004	0	0	49	0	39	0	42	0	34	M2P
Ta300-Ti600-SkC-i005	0	0	65	0	41	0	32	0	39	LNS
Ta300-Ti600-SkC-i006	0	0	144	0	35	0	33	0	42	LNS
Ta300-Ti600-SkC-i007	0	0	62	0	29	0	40	0	36	CP
Ta300-Ti600-SkC-i008	0	1	1436	0	35	0	29	0	46	LNS
Ta300-Ti600-SkC-i009	0	0	41	0	32	0	35	0	54	CP
Ta300-Ti600-SkC-i010	0	0	742	0	37	0	29	0	48	LNS
Ta300-Ti600-SkC-i011	0	0	82	0	32	0	32	0	29	M2P
Ta300-Ti600-SkC-i012	0	0	60	0	29	0	29	0	38	CP-LNS
Ta300-Ti600-SkC-i013	0	0	58	0	28	0	38	0	41	CP
Ta300-Ti600-SkC-i014	0	0	80	0	20	0	29	0	30	CP
Ta300-Ti600-SkC-i015	0	0	102	0	37	0	35	0	56	LNS
Ta300-Ti600-SkC-i016	0	0	73	0	17	0	16	0	34	LNS
Ta300-Ti600-SkC-i017	0	0	246	0	38	0	60	0	42	CP
Ta300-Ti600-SkC-i018	0	0	479	0	30	0	33	0	49	CP
Ta300-Ti600-SkC-i019	0	0	48	0	46	0	39	0	35	M2P
Ta300-Ti600-SkC-i020	0	0	94	0	32	0	31	0	40	LNS
Ta300-Ti600-SkC-i021	0	0	97	0	28	0	51	0	30	CP
Ta300-Ti600-SkC-i022	0	0	66	0	33	0	34	0	33	CP-M2P
Ta300-Ti600-SkC-i023	0	0	87	0	33	0	26	0	35	LNS
Ta300-Ti600-SkC-i024	0	0	145	0	43	0	31	0	62	LNS
Ta300-Ti600-SkC-i025	0	0	50	0	18	0	25	0	37	CP
Ta300-Ti600-SkC-i026	0	0	560	0	30	0	26	0	50	LNS
Ta300-Ti600-SkC-i027	0	0	228	0	54	0	43	0	40	M2P
Ta300-Ti600-SkC-i028	0	0	81	0	30	0	25	0	31	LNS
Ta300-Ti600-SkC-i029	0	0	74	0	20	0	22	0	33	CP
Ta300-Ti600-SkCR-i000	0	1	933	0	39	0	21	0	35	LNS
Ta300-Ti600-SkCR-i001	0	0	84	0	36	0	33	0	43	LNS
Ta300-Ti600-SkCR-i002	0	1	1911	0	28	0	29	0	44	CP
Ta300-Ti600-SkCR-i003	0	0	121	0	15	0	36	0	42	CP
Ta300-Ti600-SkCR-i004	0	0	166	0	21	0	35	0	42	CP
Ta300-Ti600-SkCR-i005	0	0	579	0	29	0	28	0	35	LNS
Ta300-Ti600-SkCR-i006	0	0	51	0	34	0	40	0	59	CP
Ta300-Ti600-SkCR-i007	0	0	53	0	32	0	34	0	46	CP
Ta300-Ti600-SkCR-i008	0	0	46	0	33	0	38	0	30	M2P
Ta300-Ti600-SkCR-i009	0	0	115	0	48	0	23	0	64	LNS
Ta300-Ti600-SkCR-i010	0	0	55	0	26	0	19	0	28	LNS
Ta300-Ti600-SkCR-i011	0	0	62	0	31	0	22	0	26	LNS
Ta300-Ti600-SkCR-i012	0	0	531	0	26	0	39	0	35	CP
Ta300-Ti600-SkCR-i013	0	0	776	0	29	0	19	0	31	LNS
Ta300-Ti600-SkCR-i014	1	2	2263	1	42	1	101	1	154	CP
Ta300-Ti600-SkCR-i015	0	0	28	0	25	0	61	0	40	CP
Ta300-Ti600-SkCR-i016	0	0	74	0	30	0	48	0	52	CP
Ta300-Ti600-SkCR-i017	0	0	53	0	35	0	38	0	22	M2P
Ta300-Ti600-SkCR-i018	0	0	71	0	25	0	34	0	42	CP
Ta300-Ti600-SkCR-i019	0	0	42	0	31	0	28	0	50	LNS
Ta300-Ti600-SkCR-i020	0	0	116	0	31	0	23	0	30	LNS
Ta300-Ti600-SkCR-i021	0	0	285	0	31	0	34	0	52	CP
Ta300-Ti600-SkCR-i022	0	0	265	0	35	0	22	0	34	LNS
Ta300-Ti600-SkCR-i023	0	0	59	0	40	0	27	0	48	LNS
Ta300-Ti600-SkCR-i024	1	1	1901	1	32	1	83	1	58	CP
Ta300-Ti600-SkCR-i025	0	0	529	0	46	0	67	0	32	M2P
Ta300-Ti600-SkCR-i026	0	0	99	0	26	0	19	0	47	LNS
Ta300-Ti600-SkCR-i027	0	0	499	0	24	0	28	0	57	CP
Ta300-Ti600-SkCR-i028	0	0	40	0	44	0	28	0	29	LNS
Ta300-Ti600-SkCR-i029	0	0	53	0	25	0	22	0	42	LNS
Ta300-Ti800-SkC-i000	0	0	100	0	36	0	16	0	23	LNS
Ta300-Ti800-SkC-i001	0	3	1194	2	41	2	129	2	143	CP
Ta300-Ti800-SkC-i002	0	2	1574	0	51	0	24	0	26	LNS
Ta300-Ti800-SkC-i003	0	1	1385	0	76	0	24	0	42	LNS
Ta300-Ti800-SkC-i004	0			0	30	0	19	0	38	LNS
Ta300-Ti800-SkC-i005	0	2	1523	1	26	1	75	1	40	CP
Ta300-Ti800-SkC-i006	0	0	168	0	40	0	19	0	44	LNS
Ta300-Ti800-SkC-i007	0	0	186	0	36	0	23	0	43	LNS
Ta300-Ti800-SkC-i008	0	0	83	0	36	0	25	0	47	LNS
Ta300-Ti800-SkC-i009	0	2	1811	0	36	0	20	0	41	LNS
Ta300-Ti800-SkC-i010	0	1	1322	0	37	0	20	0	37	LNS
Ta300-Ti800-SkC-i011	0	0	483	0	43	0	41	0	32	M2P
Ta300-Ti800-SkC-i012	0	0	524	0	47	0	19	0	29	LNS
Ta300-Ti800-SkC-i013	0	0	340	0	41	0	23	0	54	LNS

Instance	U	PLNE		PPC		LNS		M2P		Meilleur
		U	Δ	U	Δ	U	Δ	U	Δ	
Ta300-Ti800-SkC-i014	0	3	1520	0	48	0	324	1	100	CP
Ta300-Ti800-SkC-i015	0	0	102	0	38	0	22	0	32	LNS
Ta300-Ti800-SkC-i016	0	0	562	0	58	0	27	0	33	LNS
Ta300-Ti800-SkC-i017	0	4	1497	0	92	0	30	0	67	LNS
Ta300-Ti800-SkC-i018	0	0	275	0	63	0	30	0	37	LNS
Ta300-Ti800-SkC-i019	0	1	1422	0	60	0	18	0	35	LNS
Ta300-Ti800-SkC-i020	0	0	181	0	43	0	34	0	57	LNS
Ta300-Ti800-SkC-i021	0	4	1496	0	30	0	19	0	38	LNS
Ta300-Ti800-SkC-i022	0	0	31	0	51	0	26	0	38	LNS
Ta300-Ti800-SkC-i023	0	0	184	0	36	0	44	0	33	M2P
Ta300-Ti800-SkC-i024	0	0	596	0	44	0	25	0	39	LNS
Ta300-Ti800-SkC-i025	0	0	47	0	45	0	17	0	22	LNS
Ta300-Ti800-SkC-i026	0	0	184	0	39	0	16	0	30	LNS
Ta300-Ti800-SkC-i027	0	1	1465	0	38	0	17	0	31	LNS
Ta300-Ti800-SkC-i028	0	0	244	0	47	0	26	0	44	LNS
Ta300-Ti800-SkC-i029	0	0	737	0	44	0	16	0	43	LNS
Ta300-Ti800-SkCR-i000	0	0	478	0	28	0	19	0	45	LNS
Ta300-Ti800-SkCR-i001	3	4	1182	4	44	3	49	6	70	LNS
Ta300-Ti800-SkCR-i002	0	0	563	0	43	0	29	0	46	LNS
Ta300-Ti800-SkCR-i003	0	1	1685	0	49	0	33	0	58	LNS
Ta300-Ti800-SkCR-i004	0	0	283	0	65	0	17	0	41	LNS
Ta300-Ti800-SkCR-i005	0	1	1684	0	32	0	57	0	32	CP-M2P
Ta300-Ti800-SkCR-i006	0	4	1328	2	56	2	40	2	152	LNS
Ta300-Ti800-SkCR-i007	0	1	1277	0	41	0	22	0	25	LNS
Ta300-Ti800-SkCR-i008	0	0	127	0	42	0	28	0	41	LNS
Ta300-Ti800-SkCR-i009	0	1	2098	0	40	0	22	0	36	LNS
Ta300-Ti800-SkCR-i010	0	1	1321	0	39	0	25	0	33	LNS
Ta300-Ti800-SkCR-i011	0	0	347	0	29	0	28	0	32	LNS
Ta300-Ti800-SkCR-i012	0	3	1081	0	40	0	23	0	29	LNS
Ta300-Ti800-SkCR-i013	0	0	44	0	35	0	19	0	24	LNS
Ta300-Ti800-SkCR-i014	0	1	1229	0	47	0	26	0	33	LNS
Ta300-Ti800-SkCR-i015	0	0	177	0	33	0	42	0	36	CP
Ta300-Ti800-SkCR-i016	0	1	1034	0	40	0	18	0	38	LNS
Ta300-Ti800-SkCR-i017	0	5	1575	0	32	0	31	0	45	LNS
Ta300-Ti800-SkCR-i018	0	2	1067	0	47	0	18	0	37	LNS
Ta300-Ti800-SkCR-i019	0	3	1272	4	45	3	33	4	88	LNS
Ta300-Ti800-SkCR-i020	0	2	1253	0	47	0	22	0	27	LNS
Ta300-Ti800-SkCR-i021	1	1	1285	1	30	1	62	1	48	CP
Ta300-Ti800-SkCR-i022	0	3	1401	0	44	0	37	0	47	LNS
Ta300-Ti800-SkCR-i023	0	1	1237	0	33	0	32	0	45	LNS
Ta300-Ti800-SkCR-i024	0	1	1838	0	38	0	29	0	39	LNS
Ta300-Ti800-SkCR-i025	0	0	130	0	41	0	25	0	37	LNS
Ta300-Ti800-SkCR-i026	0			0	46	0	40	0	35	M2P
Ta300-Ti800-SkCR-i027	0	0	226	0	80	0	37	0	35	M2P
Ta300-Ti800-SkCR-i028	0	0	457	0	46	0	29	0	34	LNS
Ta300-Ti800-SkCR-i029	0	0	173	0	34	0	23	0	48	LNS
Ta300-Ti1000-SkC-i000	0	2	1426	0	71	0	34	0	71	LNS
Ta300-Ti1000-SkC-i001	0	1	1263	2	104	0	34	0	58	LNS
Ta300-Ti1000-SkC-i002	1	2	1977	2	56	2	58	2	72	CP
Ta300-Ti1000-SkC-i003	0	2	1325	0	108	0	64	2	176	LNS
Ta300-Ti1000-SkC-i004	0	1	1772	0	51	0	37	0	38	LNS
Ta300-Ti1000-SkC-i005	0	3	987	4	56	0	22	0	25	LNS
Ta300-Ti1000-SkC-i006	0	0	199	0	49	0	29	0	40	LNS
Ta300-Ti1000-SkC-i007	0	2	905	0	61	0	68	0	36	M2P
Ta300-Ti1000-SkC-i008	0	1	1220	1	65	0	52	0	40	M2P
Ta300-Ti1000-SkC-i009	0	4	1612	4	52	1	203	1	163	M2P
Ta300-Ti1000-SkC-i010	0	0	563	0	57	0	29	0	24	M2P
Ta300-Ti1000-SkC-i011	0	1	1521	0	53	0	18	0	30	LNS
Ta300-Ti1000-SkC-i012	2	5	1267	5	60	5	101	5	23	M2P
Ta300-Ti1000-SkC-i013	0	1	1231	0	55	0	32	0	50	LNS
Ta300-Ti1000-SkC-i014	0	2	1511	0	37	0	17	0	31	LNS
Ta300-Ti1000-SkC-i015	0	1	1585	0	45	0	20	0	60	LNS
Ta300-Ti1000-SkC-i016	0	5	1342	0	50	0	34	0	40	LNS
Ta300-Ti1000-SkC-i017	0	1	1822	0	51	0	45	0	46	LNS
Ta300-Ti1000-SkC-i018	2	6	1229	6	61	6	114	6	129	CP
Ta300-Ti1000-SkC-i019	0	7	1366	6	82	4	287	4	93	M2P
Ta300-Ti1000-SkC-i020	0	4	1530	3	43	2	88	2	115	LNS
Ta300-Ti1000-SkC-i021	0	0	381	0	61	0	37	0	46	LNS
Ta300-Ti1000-SkC-i022	0	5	1198	1	67	0	10	0	65	LNS
Ta300-Ti1000-SkC-i023	0	4	1162	1	85	0	33	0	66	LNS
Ta300-Ti1000-SkC-i024	0	0	255	0	79	0	19	0	28	LNS
Ta300-Ti1000-SkC-i025	0	0	732	0	70	0	25	0	44	LNS
Ta300-Ti1000-SkC-i026	0	0	422	0	67	0	30	1	109	LNS
Ta300-Ti1000-SkC-i027	0	1	1456	1	45	0	35	0	59	LNS

Instance	$\underline{U}$	PLNE		PPC		LNS		M2P		Meilleur
		U	Δ	U	Δ	U	Δ	U	Δ	
Ta300-Ti1000-SkC-i028	0	5	1653	8	61	2	254	7	66	LNS
Ta300-Ti1000-SkC-i029	0	3	1703	0	71	0	26	0	72	LNS
Ta300-Ti1000-SkCR-i000	0	1	1344	4	129	1	125	2	128	LNS
Ta300-Ti1000-SkCR-i001	0	2	1231	2	74	0	34	2	67	LNS
Ta300-Ti1000-SkCR-i002	3	11	1363	10	91	10	98	10	46	M2P
Ta300-Ti1000-SkCR-i003	0	1	1481	0	55	0	17	2	41	LNS
Ta300-Ti1000-SkCR-i004	0	3	1350	3	43	3	110	6	50	CP
Ta300-Ti1000-SkCR-i005	0	2	1578	0	73	0	22	0	78	LNS
Ta300-Ti1000-SkCR-i006	0	0	439	0	43	0	40	0	30	M2P
Ta300-Ti1000-SkCR-i007	0	1	1352	0	83	0	26	0	40	LNS
Ta300-Ti1000-SkCR-i008	0	1	1401	0	97	0	38	0	47	LNS
Ta300-Ti1000-SkCR-i009	0	0	238	0	52	0	21	0	104	LNS
Ta300-Ti1000-SkCR-i010	0	2	1460	0	69	0	35	0	67	LNS
Ta300-Ti1000-SkCR-i011	0	3	1327	2	105	1	67	2	81	LNS
Ta300-Ti1000-SkCR-i012	0	1	1769	3	42	0	21	0	26	LNS
Ta300-Ti1000-SkCR-i013	0	4	1516	0	88	0	33	4	195	LNS
Ta300-Ti1000-SkCR-i014	0	0	669	0	43	0	40	0	42	LNS
Ta300-Ti1000-SkCR-i015	0	0	840	0	126	0	32	0	34	LNS
Ta300-Ti1000-SkCR-i016	0	0	412	0	57	0	40	0	62	LNS
Ta300-Ti1000-SkCR-i017	0	0	103	0	67	0	15	0	34	LNS
Ta300-Ti1000-SkCR-i018	0	2	1512	0	60	0	24	0	35	LNS
Ta300-Ti1000-SkCR-i019	0	1	1406	0	46	0	36	0	64	LNS
Ta300-Ti1000-SkCR-i020	0	4	1852	1	39	0	27	0	39	LNS
Ta300-Ti1000-SkCR-i021	0	1	1425	0	49	0	31	0	27	M2P
Ta300-Ti1000-SkCR-i022	0	5	1413	7	45	5	528	6	40	LNS
Ta300-Ti1000-SkCR-i023	0	3	1275	3	63	1	136	3	44	LNS
Ta300-Ti1000-SkCR-i024	0	3	1467	1	86	0	33	0	31	M2P
Ta300-Ti1000-SkCR-i025	0	0	459	0	61	0	21	0	51	LNS
Ta300-Ti1000-SkCR-i026	0	0	119	0	44	0	13	0	28	LNS
Ta300-Ti1000-SkCR-i027	0	3	1044	0	49	0	48	0	22	M2P
Ta300-Ti1000-SkCR-i028	0	1	1247	1	51	0	29	0	30	LNS
Ta300-Ti1000-SkCR-i029	0	1	1446	0	132	0	31	0	55	LNS
Ta400-Ti600-SkC-i000	0			0	26	0	27	0	42	CP
Ta400-Ti600-SkC-i001	0			0	19	0	40	0	42	CP
Ta400-Ti600-SkC-i002	0			0	16	0	25	0	67	CP
Ta400-Ti600-SkC-i003	0			0	31	0	38	0	51	CP
Ta400-Ti600-SkC-i004	0			0	19	0	39	0	88	CP
Ta400-Ti600-SkC-i005	0			0	25	0	28	0	55	CP
Ta400-Ti600-SkC-i006	0			0	23	0	40	0	27	CP
Ta400-Ti600-SkC-i007	0			0	27	0	24	0	27	LNS
Ta400-Ti600-SkC-i008	0			0	22	0	33	0	46	CP
Ta400-Ti600-SkC-i009	0			0	20	0	32	0	42	CP
Ta400-Ti600-SkC-i010	0			0	18	0	27	0	39	CP
Ta400-Ti600-SkC-i011	0			0	32	0	26	0	34	LNS
Ta400-Ti600-SkC-i012	0			0	27	0	27	0	57	CP-LNS
Ta400-Ti600-SkC-i013	0			0	28	0	46	0	30	CP
Ta400-Ti600-SkC-i014	0			0	28	0	22	0	52	LNS
Ta400-Ti600-SkC-i015	0			0	76	0	30	0	40	LNS
Ta400-Ti600-SkC-i016	0			0	21	0	33	0	41	CP
Ta400-Ti600-SkC-i017	0			0	20	0	27	0	33	CP
Ta400-Ti600-SkC-i018	0			0	38	0	47	0	49	CP
Ta400-Ti600-SkC-i019	0			0	26	0	29	0	39	CP
Ta400-Ti600-SkC-i020	0			0	37	0	36	0	64	LNS
Ta400-Ti600-SkC-i021	0			0	31	0	44	0	45	CP
Ta400-Ti600-SkC-i022	0			0	27	0	41	0	85	CP
Ta400-Ti600-SkC-i023	0			0	23	0	18	0	31	LNS
Ta400-Ti600-SkC-i024	0			0	22	0	33	0	53	CP
Ta400-Ti600-SkC-i025	0			0	29	0	22	0	63	LNS
Ta400-Ti600-SkC-i026	0			0	27	0	31	0	45	CP
Ta400-Ti600-SkC-i027	0			0	10	0	28	0	71	CP
Ta400-Ti600-SkC-i028	0			0	41	0	48	0	56	CP
Ta400-Ti600-SkC-i029	0			0	33	0	45	0	56	CP
Ta400-Ti600-SkCR-i000	0			0	37	0	34	0	41	LNS
Ta400-Ti600-SkCR-i001	0			0	30	0	23	0	53	LNS
Ta400-Ti600-SkCR-i002	0			0	34	0	40	0	64	CP
Ta400-Ti600-SkCR-i003	0			0	33	0	22	0	31	LNS
Ta400-Ti600-SkCR-i004	0			0	20	0	38	0	42	CP
Ta400-Ti600-SkCR-i005	0			0	31	0	42	0	36	CP
Ta400-Ti600-SkCR-i006	0			0	36	0	42	0	29	M2P
Ta400-Ti600-SkCR-i007	1			1	35	1	71	1	104	CP
Ta400-Ti600-SkCR-i008	0			0	25	0	38	0	61	CP
Ta400-Ti600-SkCR-i009	0			0	18	0	42	0	41	CP
Ta400-Ti600-SkCR-i010	0			0	18	0	35	0	26	CP
Ta400-Ti600-SkCR-i011	0			0	38	0	42	0	41	CP

Instance	U	PLNE		PPC		LNS		M2P		Meilleur
		U	Δ	U	Δ	U	Δ	U	Δ	
Ta400-Ti600-SkCR-i012	0			0	31	0	39	0	42	CP
Ta400-Ti600-SkCR-i013	0			0	32	0	37	0	34	CP
Ta400-Ti600-SkCR-i014	0			0	21	0	35	0	34	CP
Ta400-Ti600-SkCR-i015	0			0	37	0	32	0	39	LNS
Ta400-Ti600-SkCR-i016	0			0	29	0	31	0	31	CP
Ta400-Ti600-SkCR-i017	0			0	31	0	24	0	38	LNS
Ta400-Ti600-SkCR-i018	0			0	29	0	34	0	49	CP
Ta400-Ti600-SkCR-i019	0			0	23	0	29	0	44	CP
Ta400-Ti600-SkCR-i020	0			0	31	0	47	0	54	CP
Ta400-Ti600-SkCR-i021	0			0	44	0	30	0	66	LNS
Ta400-Ti600-SkCR-i022	0			0	33	0	41	0	70	CP
Ta400-Ti600-SkCR-i023	0			0	38	0	23	0	37	LNS
Ta400-Ti600-SkCR-i024	0			0	37	0	27	0	38	LNS
Ta400-Ti600-SkCR-i025	0			0	21	0	41	0	84	CP
Ta400-Ti600-SkCR-i026	0			0	39	0	36	0	58	LNS
Ta400-Ti600-SkCR-i027	0			0	38	0	27	0	42	LNS
Ta400-Ti600-SkCR-i028	0			0	33	0	31	0	47	LNS
Ta400-Ti600-SkCR-i029	0			0	32	0	31	0	44	LNS
Ta400-Ti800-SkC-i000	0			0	29	0	36	0	27	M2P
Ta400-Ti800-SkC-i001	0			0	34	0	31	0	57	LNS
Ta400-Ti800-SkC-i002	0			0	33	0	18	0	39	LNS
Ta400-Ti800-SkC-i003	0			0	28	0	52	0	30	CP
Ta400-Ti800-SkC-i004	0			0	33	0	19	0	19	LNS-M2P
Ta400-Ti800-SkC-i005	0			0	42	0	31	0	47	LNS
Ta400-Ti800-SkC-i006	0			0	35	0	51	0	45	CP
Ta400-Ti800-SkC-i007	0			0	33	0	27	0	40	LNS
Ta400-Ti800-SkC-i008	0			0	39	0	30	0	38	LNS
Ta400-Ti800-SkC-i009	0			0	41	0	48	1	138	CP
Ta400-Ti800-SkC-i010	0			0	38	0	29	0	32	LNS
Ta400-Ti800-SkC-i011	0			0	34	0	24	0	30	LNS
Ta400-Ti800-SkC-i012	0			0	24	0	28	0	29	CP
Ta400-Ti800-SkC-i013	0			0	30	0	19	0	72	LNS
Ta400-Ti800-SkC-i014	0			0	38	0	36	0	36	LNS-M2P
Ta400-Ti800-SkC-i015	0			0	39	0	33	0	37	LNS
Ta400-Ti800-SkC-i016	0			0	46	0	23	0	39	LNS
Ta400-Ti800-SkC-i017	0			0	35	0	28	0	25	M2P
Ta400-Ti800-SkC-i018	0			0	45	0	24	0	36	LNS
Ta400-Ti800-SkC-i019	0			0	37	0	30	0	79	LNS
Ta400-Ti800-SkC-i020	0			0	44	0	25	0	45	LNS
Ta400-Ti800-SkC-i021	0			0	20	0	18	0	38	LNS
Ta400-Ti800-SkC-i022	0			0	32	0	31	0	54	LNS
Ta400-Ti800-SkC-i023	0			0	41	0	14	2	111	LNS
Ta400-Ti800-SkC-i024	0			0	36	0	19	0	38	LNS
Ta400-Ti800-SkC-i025	0			0	31	0	22	0	42	LNS
Ta400-Ti800-SkC-i026	0			0	24	0	51	0	36	CP
Ta400-Ti800-SkC-i027	0			0	38	0	17	0	41	LNS
Ta400-Ti800-SkC-i028	0			0	39	0	24	0	39	LNS
Ta400-Ti800-SkC-i029	0			0	41	0	24	0	38	LNS
Ta400-Ti800-SkCR-i000	0			0	46	0	38	0	49	LNS
Ta400-Ti800-SkCR-i001	0			0	34	0	35	0	34	CP-M2P
Ta400-Ti800-SkCR-i002	0			0	36	0	46	0	30	M2P
Ta400-Ti800-SkCR-i003	0			0	39	0	36	0	30	M2P
Ta400-Ti800-SkCR-i004	0			0	61	0	24	0	53	LNS
Ta400-Ti800-SkCR-i005	0			0	42	0	41	0	51	LNS
Ta400-Ti800-SkCR-i006	0			0	40	0	21	0	66	LNS
Ta400-Ti800-SkCR-i007	0			0	45	0	23	0	32	LNS
Ta400-Ti800-SkCR-i008	0			1	28	1	72	1	68	CP
Ta400-Ti800-SkCR-i009	0			0	40	0	39	0	43	LNS
Ta400-Ti800-SkCR-i010	0			0	42	0	28	0	36	LNS
Ta400-Ti800-SkCR-i011	0			0	36	0	29	0	49	LNS
Ta400-Ti800-SkCR-i012	0			0	43	0	40	0	34	M2P
Ta400-Ti800-SkCR-i013	0			0	35	0	29	0	53	LNS
Ta400-Ti800-SkCR-i014	0			0	44	0	25	0	24	M2P
Ta400-Ti800-SkCR-i015	0			0	39	0	35	0	26	M2P
Ta400-Ti800-SkCR-i016	1			1	39	1	115	1	130	CP
Ta400-Ti800-SkCR-i017	0			0	39	0	35	0	60	LNS
Ta400-Ti800-SkCR-i018	0			1	54	0	40	1	64	LNS
Ta400-Ti800-SkCR-i019	0			0	88	0	42	0	92	LNS
Ta400-Ti800-SkCR-i020	0			0	33	0	28	0	23	M2P
Ta400-Ti800-SkCR-i021	0			0	79	0	26	0	39	LNS
Ta400-Ti800-SkCR-i022	0			0	38	0	42	0	35	M2P
Ta400-Ti800-SkCR-i023	0			0	41	0	36	0	39	LNS
Ta400-Ti800-SkCR-i024	0			0	30	0	27	0	25	M2P
Ta400-Ti800-SkCR-i025	0			0	33	0	31	0	57	LNS

Instance	U	PLNE		PPC		LNS		M2P		Meilleur
		U	Δ	U	Δ	U	Δ	U	Δ	
Ta400-Ti800-SkCR-i026	0			0	58	0	34	0	40	LNS
Ta400-Ti800-SkCR-i027	0			0	49	0	43	0	122	LNS
Ta400-Ti800-SkCR-i028	0			0	35	0	35	0	55	CP-LNS
Ta400-Ti800-SkCR-i029	0			0	37	0	32	0	86	LNS
Ta400-Ti1000-SkC-i000	0			0	53	0	35	0	68	LNS
Ta400-Ti1000-SkC-i001	0			0	57	0	50	0	108	LNS
Ta400-Ti1000-SkC-i002	0			0	44	0	45	0	30	M2P
Ta400-Ti1000-SkC-i003	0			0	64	0	24	0	73	LNS
Ta400-Ti1000-SkC-i004	0			0	44	0	48	0	60	CP
Ta400-Ti1000-SkC-i005	0			3	43	1	70	2	117	LNS
Ta400-Ti1000-SkC-i006	0			1	63	0	42	0	16	M2P
Ta400-Ti1000-SkC-i007	0			0	42	0	29	0	38	LNS
Ta400-Ti1000-SkC-i008	0			0	62	0	36	0	36	LNS-M2P
Ta400-Ti1000-SkC-i009	0			0	62	0	39	0	59	LNS
Ta400-Ti1000-SkC-i010	0			0	46	0	18	0	44	LNS
Ta400-Ti1000-SkC-i011	0			0	45	0	27	0	35	LNS
Ta400-Ti1000-SkC-i012	0			0	46	0	32	0	56	LNS
Ta400-Ti1000-SkC-i013	0			0	47	0	49	0	51	CP
Ta400-Ti1000-SkC-i014	0			0	49	0	49	0	39	M2P
Ta400-Ti1000-SkC-i015	0			0	53	0	27	0	37	LNS
Ta400-Ti1000-SkC-i016	0			0	48	0	31	0	64	LNS
Ta400-Ti1000-SkC-i017	0			0	77	0	26	0	60	LNS
Ta400-Ti1000-SkC-i018	0			4	43	3	61	3	135	LNS
Ta400-Ti1000-SkC-i019	0			0	63	0	35	0	40	LNS
Ta400-Ti1000-SkC-i020	0			2	65	1	96	1	60	M2P
Ta400-Ti1000-SkC-i021	0			0	55	0	23	0	62	LNS
Ta400-Ti1000-SkC-i022	0			0	45	0	31	0	49	LNS
Ta400-Ti1000-SkC-i023	0			0	31	0	28	0	41	LNS
Ta400-Ti1000-SkC-i024	1			3	46	1	89	5	224	LNS
Ta400-Ti1000-SkC-i025	0			4	53	3	319	6	140	LNS
Ta400-Ti1000-SkC-i026	0			7	39	6	114	6	87	M2P
Ta400-Ti1000-SkC-i027	0			0	41	0	39	0	100	LNS
Ta400-Ti1000-SkC-i028	0			0	77	0	60	0	32	M2P
Ta400-Ti1000-SkC-i029	0			0	47	0	55	0	47	CP-M2P
Ta400-Ti1000-SkCR-i000	0			1	68	0	19	2	213	LNS
Ta400-Ti1000-SkCR-i001	0			2	62	2	74	3	103	CP
Ta400-Ti1000-SkCR-i002	0			0	58	0	41	0	65	LNS
Ta400-Ti1000-SkCR-i003	0			0	44	0	35	0	42	LNS
Ta400-Ti1000-SkCR-i004	0			2	58	0	52	0	115	LNS
Ta400-Ti1000-SkCR-i005	0			0	40	0	18	0	50	LNS
Ta400-Ti1000-SkCR-i006	0			5	49	3	100	4	134	LNS
Ta400-Ti1000-SkCR-i007	0			0	36	0	102	0	45	CP
Ta400-Ti1000-SkCR-i008	1			3	46	1	183	2	93	LNS
Ta400-Ti1000-SkCR-i009	2			12	48	11	290	13	168	LNS
Ta400-Ti1000-SkCR-i010	0			4	46	2	157	4	105	LNS
Ta400-Ti1000-SkCR-i011	0			0	49	0	25	0	46	LNS
Ta400-Ti1000-SkCR-i012	0			6	67	3	217	5	122	LNS
Ta400-Ti1000-SkCR-i013	0			1	59	0	112	1	112	LNS
Ta400-Ti1000-SkCR-i014	2	7	1320	4	65	2	86	5	310	LNS
Ta400-Ti1000-SkCR-i015	0			3	57	1	129	3	110	LNS
Ta400-Ti1000-SkCR-i016	0			0	63	0	23	1	94	LNS
Ta400-Ti1000-SkCR-i017	0			5	44	3	83	5	74	LNS
Ta400-Ti1000-SkCR-i018	0			6	61	5	416	6	267	LNS
Ta400-Ti1000-SkCR-i019	0			0	43	0	39	0	43	LNS
Ta400-Ti1000-SkCR-i020	0			1	59	0	62	0	70	LNS
Ta400-Ti1000-SkCR-i021	0			0	53	0	43	0	51	LNS
Ta400-Ti1000-SkCR-i022	0			0	37	0	24	0	32	LNS
Ta400-Ti1000-SkCR-i023	0			0	51	0	34	0	39	LNS
Ta400-Ti1000-SkCR-i024	0			0	84	0	36	0	52	LNS
Ta400-Ti1000-SkCR-i025	0			0	67	0	45	0	29	M2P
Ta400-Ti1000-SkCR-i026	0			0	53	0	26	0	29	LNS
Ta400-Ti1000-SkCR-i027	0			0	49	0	54	2	209	CP
Ta400-Ti1000-SkCR-i028	0			0	50	0	31	0	38	LNS
Ta400-Ti1000-SkCR-i029	0			0	56	0	20	0	34	LNS

Publications scientifiques

Sommaire

C.1 Journaux internationaux

Jean-Guillaume Fages et Tanguy Lapègue : *Filtering atmostnvalue with difference constraints : Application to the shift minimisation personnel task scheduling problem* In **Artificial Intelligence**, 2014, Volume 212, Pages 116 - 133

Tanguy Lapègue, Odile Bellenguez-Morineau et Damien Prot : *A constraint-based approach for the shift design personnel task scheduling problem with equity* In **Computers & Operations Research**, 2013, Volume 40, Issue 10, Pages 2450-2465

C.2 Conférences internationales avec actes

Tanguy Lapègue, Odile Bellenguez-Morineau et Damien Prot : *A tour scheduling problem with fixed jobs : use of constraint programming* In Proceedings of Practice and Theory of Automated Timetabling 2012 (**PATAT 2012**)

Tanguy Lapègue, Damien Prot et Odile Bellenguez-Morineau : *A large neighborhood search for the Shift Design Personnel Task Scheduling Problem with Equity* In Proceedings of the Multidisciplinary International Conference on Scheduling : Theory and Applications 2013 (**MISTA 2013**)

Jean-Guillaume Fages et Tanguy Lapègue : *Filtering atmostnvalue with difference constraints : Application to the shift minimisation personnel task scheduling problem* In The 19th International Conference on Principles and Practice of Constraint Programming (**CP 2013**), editor : Christian Schulte,

Volume 8124 of Lecture Notes in Computer Science, Pages 63-79. Springer Berlin Heidelberg, 2013 (*Best student paper*)

C.3 Conférences nationales

Tanguy Lapègue, Odile Bellenguez-Morineau et Damien Prot : *Ordonnancement de personnel avec affectation de tâches : un modèle mathématique* In ROADEF : SOCIÉTÉ FRANÇAISE DE RECHERCHE OPÉRATIONNELLE ET D'AIDE À LA DÉCISION, (**ROADEF 2013**)

Tanguy Lapègue, Odile Bellenguez-Morineau et Damien Prot : *Dimensionnement d'équipe pour la réalisation d'étude de produits pharmaceutiques* In ROADEF : SOCIÉTÉ FRANÇAISE DE RECHERCHE OPÉRATIONNELLE ET D'AIDE À LA DÉCISION, (**ROADEF 2014**)

C.4 Soumissions en cours

Tanguy Lapègue, Odile Bellenguez-Morineau et Damien Prot : *Overview on methods for the Shift Design and Personnel Task Scheduling Problem with Equity objective* In 10th International Conference on Modeling, Optimization & SIMulation (**MOSIM 2014**)

Damien Prot, Tanguy Lapègue et Odile Bellenguez-Morineau : *A two phase method for the Shift Design Personnel Task Scheduling Problem with equity objective* In **International Journal of Production Research**

Bibliographie

[1] Abdennadher, S., Schlenker, H. : Nurse scheduling using constraint logic programming. In : Innovative Applications of Artificial Intelligence Conference. pp. 838–843 (1999) 96

[2] Aickelin, U., Dowsland, K.A. : Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem. Journal of Scheduling 3(3), 139–153 (2000) 41

[3] Aickelin, U., Dowsland, K.A. : An Indirect Genetic Algorithm for a Nurse Scheduling Problem An Indirect Genetic Algorithm for a Nurse Scheduling Problem. Computers & Operations Research 31(5), 761–778 (2003) 41

[4] Aickelin, U., White, P. : Building Better Nurse Scheduling Algorithms. Annals of Operations Research 128(1-4), 159–177 (2004) 41

[5] Alfares, H.K. : Aircraft maintenance workforce scheduling : A case study. Journal of Quality in Maintenance Engineering 5(2), 78–89 (1999) 43

[6] Alfares, H.K. : Survey, Categorization, and Comparison of Recent Tour Scheduling Literature. Annals of Operations Research 127, 145–175 (2004) 38, 42

[7] Arkin, E.M., Sivlerberg, E.B. : Scheduling jobs with fixed start and end times. Discrete Applied Mathematics 18(1), 1–8 (1987) 52, 57

[8] Ashley, D.W. : A spreadsheet optimization system for library staff scheduling. Computers & Operations Research 22(6), 615–624 (1995) 43

[9] Baker, K.R., Magazine, M.J. : Workforce Scheduling with Cyclic Demands and Day-Off Constraints. Management Science 24(2), 161–167 (1977) 49

[10] Baker, K.R. : Workforce allocation in cyclical scheduling problems : A survey. Operational Research Quarterly 27, 155–167 (1976) 38

[11] Balakrishnan, N., Wong, R.T. : A network model for the rotating workforce scheduling problem. Networks 20(1), 25–42 (1990) 43

[12] Baptiste, P., Giard, V., Haït, A., Soumis, F. : Gestion de production et ressources humaines. Presses Internationales Polytechnique (2005) 37

[13] Barcia, P., Cerdeira, J.O. : The k-track assignment problem on partial orders. Journal of Scheduling 8(2), 135–143 (2005) 51, 52, 57

[14] Barnhart, C., Hatay, L., Johnson, E.L. : Deadhead selection for the long-haul crew pairing problem. Operations Research 43(3), 491–499 (1995) 47

[15] Barnhart, C., Shenoi, R.G. : An approximate model and solution approach for the long-haul crew pairing problem. Transportation Science 32(3), 221–231 (1998) 39, 47

[16] Bartholdi, J.J. : A guaranteed-accuracy round-off algorithm for cyclic scheduling and set covering. Operations Research 29(3), 501–510 (1981) 43, 49

[17] Bartholdi, J.J., Orlin, J.B., Ratliff, H.D. : Cyclic scheduling via integer programs with circular ones. Operations Research 28(5), 1074–1085 (1980) 49

[18] Beaumont, N. : Scheduling staff using mixed integer programming. European Journal of Operational Research 98(3), 473–484 (1997) 43

[19] Bechtold, S.E., Brusco, M.J., Showalter, M.J. : A Comparative Evaluation of Labor Tour Scheduling Methods. Decision Sciences 22(4), 683–699 (1991) 38

[20] Bechtold, S.E., Jacobs, L.W. : Implicit modeling of flexible break assignments in optimal schift scheduling. Management Science 36(11) (1990) 38

[21] Bekki, Ö.B., Azizoğlu, M. : Operational fixed interval scheduling problem on uniform parallel machines. International Journal of Production Economics 112(2), 756–768 (2008) 52

[22] Beldiceanu, N. : Pruning for the minimum constraint family and for the number of distinct values constraint family. In : CP. Lecture Notes in Computer Science, vol. 2239, pp. 211–224. Springer (2001) 148, 153

[23] Beldiceanu, N., Carlsson, M., Flener, P., Pearson, J. : On the reification of global constraints. Constraints 18(1), 1–6 (2013) 155

[24] Bellanti, F. : A greedy-based neighborhood search approach to a nurse rostering problem. European Journal of Operational Research 153(1), 28–40 (2004) 40

[25] Bellenguez-Morineau, O., Néron, E. : A tabu search procedure for the multi-skill project scheduling problem. Journal of Operations and Logistics (2011) 82

[26] Bertels, S., Fahle, T. : A hybrid setup for a hybrid scenario : combining heuristics for the home health care problem. Computers & Operations Research 33(10), 2866–2890 (2006) 96

[27] Bessière, C., Hebrard, E., Hnich, B., Kiziltan, Z., Walsh, T. : Filtering Algorithms for the NValue Constraint. Constraints 11(4), 271–293 (2006) 147, 148, 149, 152, 153, 164

[28] Bilgin, B., De Causmaecker, P., Rossie, B., Berghe, G.V. : Local Search Neighbourhoods to Deal with a Novel Nurse Rostering Model. Annals of Operations Research 194(1), 33–57 (2012) 41

[29] Biotrial : Biotrial, centre de recherche biomédicale (2014), http://www.biotrial.fr/ 21

[30] Boussemart, F., Hemery, F., Lecoutre, C., Sais, L. : Boosting systematic search by weighting constraints. In : 16th European Conference on Artificial Intelligence. vol. 16, p. 146 (2004) 115

[31] Bouveret, S. : Allocation et partage équitables de ressources indivisibles : modélisation, complexité et algorithmique. Ph.D. thesis, Université de Toulouse (2007) 48

[32] Bouzina, K.I., Emmons, H. : Interval scheduling on identical machines. Journal of Global Optimization 9(3-4), 379–393 (1996) 52, 57

[33] Boyer, V., Gendron, B., Rousseau, L.M. : A branch-and-price algorithm for the multi-activity multi-task shift scheduling problem. Journal of Scheduling 17(2), 185–197 (2014) 57

[34] Bron, C., Kerbosch, J. : Algorithm 457 : finding all cliques of an undirected graph. Communications of the ACM 16(9), 575–577 (1973) 153

[35] Brucker, P., Burke, E.K., Curtois, T., Qu, R., Vanden Berghe, G. : A shift sequence based approach for nurse scheduling and a new benchmark dataset. Journal of Heuristics 16(4), 559–573 (2008) 41, 49, 57, 78

[36] Brucker, P., Nordmann, L. : The k-track assignment problem. Computing 52(2), 97–122 (1994) 52

[37] Brucker, P., Qu, R. : Network flow models for intraday personnel scheduling problems. Annals of Operations Research 218, 1–8 (2012) 43, 44, 57

[38] Brucker, P., Qu, R., Burke, E., Post, G. : A decomposition, construction and post-processing approach for nurse rostering. In : Multidisciplinary International Conference on Scheduling : Theory and Applications. pp. 397–406 (2005) 40

[39] Brusco, M.J., Jacobs, L.W. : A simulated annealing approach to the solution of flexible labour scheduling problems. Journal of the Operational Research Society 44(12), 1191–1200 (1993) 43, 57

[40] Brusco, M.J., Jacobs, L.W. : Cost analysis of alternative formulations for personnel scheduling in continuously operating organizations. European Journal of Operational Research 86(2), 249–261 (1995) 43, 57

[41] Brusco, M.J., Jacobs, L.W., Bongiorno, R.J., Lyons, D.V., Tang, B. : Improving personnel scheduling at airline stations. Operations Research 43(5), 741–751 (1995) 44, 57

[42] Brusco, M.J., Johns, T.R. : An integrated approach to shift-starting time selection and tour-schedule construction. Journal of the Operational Research Society 62, 1357–1364 (2011) 43

[43] Burke, E.K., Cowling, P. : A Memetic Approach to the Nurse Rostering Problem. Applied Intelligence 15(3), 199–214 (2001) 40, 41, 57

[44] Burke, E.K., Curtois, T., Post, G., Qu, R., Veltman, B. : A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem. European Journal of Operational Research 188(2), 330–341 (2008) 40

[45] Burke, E.K., De Causmaecker, P., Berghe, G.V. : A Hybrid Tabu Search Algorithm for the Nurse Rostering Problem. In : Proceedings of the Second Asia-Pacific Conference on Simulated Evolution and Learning. pp. 187–194 (1998) 41

[46] Burke, E.K., De Causmaecker, P., Berghe, G.V., Landeghem, H.V. : The state of the art of nurse rostering. Journal of Scheduling 7, 441–499 (2004) 39, 40

[47] Burke, E.K., De Causmaecker, P., Petrovic, S., Berghe, G.V. : Fitness evaluation for nurse scheduling problems. Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546) 2, 1139–1146 (2001) 41

[48] Burke, E.K., De Causmaecker, P., Petrovic, S., Berghe, G.V. : Variable Neighbourhood Search for Nurse Rostering Problems. In : Metaheuristics : computer decision-making, pp. 153–172. Springer (2004) 41, 49

[49] Burke, E.K., Li, J., Qu, R. : A hybrid model of integer programming and variable neighbourhood search for highly-constrained nurse rostering problems. European Journal of Operational Research 203(2), 484–493 (2010) 40, 57

[50] Cai, X., Li, K.N. : A genetic algorithm for scheduling staff of mixed skills under multi-criteria. European Journal Of Operational Research 125, 359–369 (2000) 43

[51] Caprara, A., Focacci, F., Lamma, E., Mello, P., Milano, M., Toth, P., Vigo, D. : Integrating constraint logic programming and operations research techniques for the crew rostering problem. Software Practice and Experience 28(1), 49–76 (1998) 47, 57, 96

[52] Caprara, A., Monaci, M., Toth, P. : A global method for crew planning in railway applications. In : Computer-Aided Scheduling of Public Transport, pp. 17–36. Springer (2001) 56, 57

[53] Caprara, A., Toth, P., Vigo, D., Fischetti, M. : Modeling and solving the crew rostering problem. Operations Research 46(6), 820–830 (1998) 47, 50, 57

[54] Carlisle, M.C., Lloyd, E.L. : On the k-coloring of intervals. In : Advances in Computing and Information-ICCI'91, pp. 90–101. Springer (1991) 52, 57

[55] Carter, M.W., Tovey, C.A. : When is the classroom assignment problem hard ? Operations Research 40(1), 28–39 (1992) 50, 51

[56] Cezik, T., Günlük, O., Luss, H. : An Integer Programming Model for the Weekly Tour Scheduling Problem. Naval Research Logistics 48(7), 1–27 (2001) 38, 43

[57] Chabrier, A. : A cooperative CP and LP optimizer approach for the pairing generation problem. In : International Workshop on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (1999) 39, 47, 96

[58] Chan, P., Weil, G. : Cyclical staff scheduling using constraint logic programming. In : Practice and Theory of Automated Timetabling III, pp. 159–175. Springer (2001) 43

[59] Cheang, B. : Nurse rostering problems-a bibliographic survey. European Journal of Operational Research 151(3), 447–460 (2003) 40

[60] Cheng, B.M.W., Lee, J.H.M., Wu, J.C.K. : A Nurse Rostering System Using Constraint Programming and Redundant Modeling. IEEE Transactions in Information Technology in Biomedicine 1, 44–54 (1997) 41, 49, 96

[61] Chiarandini, M., Schaerf, A., Tiozzo, F. : Solving employee timetabling problems with flexible workload using tabu search. In : Proceedings of the 3 rd Int. Conf. on the Practice and Theory of Automated Timetabling (E. Burke & W. Erben, Eds.) pp. pp. 298–302 (2000) 39

[62] CHOCO Team : choco : an Open Source Java Constraint Programming Library. Tech. rep., Ecole des Mines de Nantes (2010) 102, 116, 155, 173

[63] Chu, S.C. : Generating, scheduling and rostering of shift crew-duties : Applications at the Hong Kong International Airport. European Journal of Operational Research 177(3), 1764–1778 (2007) 39, 44

[64] Cipriano, R., Di Gaspero, L., Dovier, A. : Hybrid approaches for rostering : A case study in the integration of constraint programming and local search. In : Hybrid Metaheuristics, pp. 110–123. Springer (2006) 96

[65] Clausen, T. : Airport ground staff scheduling. DTU Management Engineering (2011) 39

[66] Costa, M.C., Jarray, F., Picouleau, C. : An acyclic days-off scheduling problem. 4OR 4(1), 73–85 (2006) 38

[67] Dahmen, S., Rekik, M. : Solving multi-activity multi-day shift scheduling problems with a hybrid heuristic. Journal of Scheduling pp. 1–17 (2014) 57

[68] Danna, E., Perron, L. : Structured vs . Unstructured Large Neighborhood Search : A Case Study on Job-Shop Scheduling Problems with Earliness and Tardiness Costs. In : Principles and Practice of Constraint Programming (CP 2003). pp. 817–821 (2003) 109

[69] Dawid, H., König, J., Strauss, C. : An enhanced rostering model for airline crews. Computers & Operations Research 28(7), 671–688 (2001) 47, 57

[70] Debruyne, R., Bessiere, C. : Some practicable filtering techniques for the constraint satisfaction problem. In : In Proceedings of the International Joint Conference on Artificial Intelligence. pp. 412–417 (1997) 151

[71] Demassey, S., Pesant, G., Rousseau, L.M. : Constraint programming based column generation for employee timetabling. In : Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, pp. 140–154. Springer (2005) 39

[72] Dijkstra, M.C., Kroon, L.G., Salomon, M., Van Nunen, J.A.E.E., van Wassenhove, L.N. : Planning the Size and Organization of KLM's Aircraft Maintenance Personnel. Interfaces 24(6), 47–58 (1994) 44, 57

[73] Dijkstra, M.C., Kroon, L.G., Van Nunen, J.A.E.E., Salommon, M. : A DSS for capacity planning of aircraft maintenance personnel. International Journal of Production Economics 23(1-3), 69–78 (1991) 44

[74] Dondeti, V., Emmons, H. : Fixed job scheduling with two types of processors. Operations Research 40(1-supplement-1), S76–S85 (1992) 54, 57

[75] Dondeti, V., Emmons, H. : Algorithms for preemptive scheduling of different classes of processors to do jobs with fixed times. European Journal of Operational Research 70(3), 316–326 (1993) 54, 57

[76] Dowling, D., Krishnamoorthy, M., Mackenzie, H., Sier, D. : Staff rostering at a large international airport. Annals of Operations Research 72, 125–147 (1997) 44, 57

[77] Dowsland, K.A. : Nurse scheduling with tabu search and strategic oscillation. European Journal of Operational Research 106(2-3), 393–407 (1998) 40, 41

[78] Edie, L.C. : Traffic delays at toll booths. Journal of the Operations Research Society of America 2(2), 107–138 (1954) 38

[79] Ehrgott, M., Gandibleux, X. : Multiobjective combinatorial optimization - theory, methodology, and applications. In : Multiple criteria optimization : State of the art annotated bibliographic surveys, pp. 369–444. Springer (2002) 168

[80] Elahipanah, M. : Task Scheduling and Activity Assignment to Work Shifts with Schedule Flexibility and Employee Preference Satisfaction. Ph.D. thesis, École Polytechnique de Montréal (2012) 57

[81] Ernst, A.T., Jiang, H., Krishnamoorthy, M., Sier, D. : Staff scheduling and rostering : A review of applications, methods and models. European Journal of Operational Research 153(1), 3–27 (2004) 38, 39

[82] Ernst, A.T., Jiang, H., Krishnamoorthy, M., Nott, H., Sier, D. : An integrated optimization model for train crew management. Annals of Operations Research 108(1-4), 211–224 (2001) 56, 57

[83] Ernst, A.T., Jiang, H., Krishnamoorthy, M., Owens, B., Sier, D. : An annotated bibliography of personnel scheduling and rostering. Annals of Operations Research 127(1-4), 21–144 (2004) 38, 49

[84] Eveborn, P., Rönnqvist, M. : Scheduler - a system for staff planning. Annals of Operations Research 128(1-4), 21–45 (2004) 43, 49, 57

[85] Fages, F. : On the use of constraint reification within search. personal communication (2013) 155

[86] Fages, J.G., Lapègue, T. : Filtering AtMostNValue with Difference Constraints : Application to the Shift Minimisation Personnel Task Scheduling Problem. In : Principles and Practice of Constraint Programming. pp. 63–79. Springer (2013) 54

[87] Fages, J.G., Lapègue, T. : Filtering AtMostNValue with difference constraints : Application to the shift minimisation personnel task scheduling problem. Artificial Intelligence 212(0), 116–133 (2014) 51, 54, 57, 166

[88] Fages, J.G., Prud'homme, C. : Parallel search within choco. personal communication (2013) 105

[89] Felici, G., Gentile, C. : A Polyhedral Approach for the Staff Rostering Problem. Management Science 50(3), 381–393 (2004) 43, 57

[90] Fischetti, M., Martello, S., Toth, P. : The fixed job schedule problem with spread-time constraints. Operations Research 35(6), 849–858 (1987) 50, 54, 57

[91] Fischetti, M., Martello, S., Toth, P. : The fixed job schedule problem with working-time constraints. Operations Research 37(3), 395–403 (1989) 54, 57

[92] Flavia, B., Guillermo, D., Javier, M. : Exploring the complexity boundary between coloring and list-coloring. Electronic Notes in Discrete Mathematics 25, 41–47 (2006) 146

[93] Flener, P., Pearson, J., Sellmann, M., Hentenryck, P.V., Ågren, M. : Dynamic structural symmetry breaking for constraint satisfaction problems. Constraints 14(4), 506–538 (2009) 154

[94] Focacci, F., Lamma, E., Mello, P., Milano, M. : A Constraint Logic Programming Approach to the Crew Rostering Problem (Technical Report) 57, 96

[95] Franses, P., Post, G. : Personnel scheduling in laboratories. In : Practice and Theory of Automated Timetabling IV, pp. 113–119. Springer (2003) 57

[96] Freling, R., Lentink, R.M., Wagelmans, A.P. : A decision support system for crew planning in passenger transportation using a flexible branch-and-price algorithm. Annals of Operations Research 127(1-4), 203–222 (2004) 47, 56, 57

[97] Garey, M.R., Johnson, D.S. : Computers and Intractability : A Guide to the Theory of NP-Completeness. W. H. Freeman (1979) 149

[98] Glover, F., McMillan, C. : The general employee scheduling problem : an integration of MS and AI. Computer & Operations Research 13(5), 563–573 (1986) 43

[99] Godard, D., Laborie, P., Nuijten, W. : Randomized Large Neighborhood Search for Cumulative Scheduling. In : International Conference on Automated Planning & Scheduling (ICAPS). pp. 81–89 (2005) 108

[100] Golumbic, M.C. : Algorithmic Graph Theory and Perfect Graphs, vol. 57. Elsevier, Annals of Discrete Mathematics, 2nd edn. (2004) 50, 60

[101] Gondran, M., Minoux, M., Vajda, S. : Graphs and Algorithms. John Wiley & Sons, Inc., New York, NY, USA (1984) 64, 146

[102] Gopalakrishnan, B., Johnson, E.L. : Airline Crew Scheduling : State-of-the-Art. Annals of Operations Research 140, 305–337 (2005) 39, 49

[103] Gopalakrishnan, M., Gopalakrishnan, S., Miller, D.M. : A decision support system for scheduling personnel in a newspaper publishing environment. Interfaces 23(4), 104–115 (1993) 43

[104] Gualandi, S., Malucelli, F. : Exact solution of graph coloring problems via constraint programming and column generation. INFORMS J. Comput. Sc. 24, 81–100 (2012) 50

[105] Halldórsson, M.M., Radhakrishnan, J. : Greed is good : Approximating independent sets in sparse and bounded-degree graphs. Algorithmica 18(1), 145–163 (1997) 149

[106] Haralick, R.M., Elliott, G.L. : Increasing tree search efficiency for constraint satisfaction problems. In : Proceedings of the 6th international joint conference on Artificial intelligence - Volume 1. pp. 356–364. IJCAI'79, Morgan Kaufmann Publishers Inc. (1979) 154

[107] He, F., Qu, R. : A constraint programming based column generation approach to nurse rostering problems. Computers & Operations Research 39(12), 3331–3343 (2012) 96

[108] Henderson, W.B., Berry, W.L. : Heuristic methods for telephone operator shift scheduling : an experimental analysis. Management Science 22(12), 1372–1380 (1976) 38

[109] Herbers, J. : Models and algorithms for ground staff scheduling on airports. Ph.D. thesis, RWTH Aachen University (2005) 44, 96

[110] Hojati, M., Patil, A.S. : An integer linear programming-based heuristic for scheduling heterogeneous, part-time service employees. European Journal of Operational Research 209(1), 37–50 (2011) 43, 57

[111] Holloran, T.J., Byrn, J.E. : United Airlines Station Manpower Planning System. Interfaces 16(1), 39–50 (1986) 44, 57

[112] Huang, Q., Lloyd, E. : Cost Constrained Fixed Job Scheduling. Lecture Notes in Computer Science 2841, 111–124 (2003) 54, 57

[113] Hung, R. : Shiftwork scheduling algorithms with phase-delay feature. International Journal of Production Research (7), 1961–1968 (1997) 43

[114] ILOG CPLEX : ILOG CPLEX 12.4. Tech. rep., IBM (2011) 75, 155

[115] Jarrah, A.I.Z., Bard, J.F., DeSilva, A.H. : Solving Large-scale Tour Scheduling Problems. Management Science 40(9), 1124–1144 (1994) 43

[116] Jaumard, B., Semet, F., Vovor, T. : A generalized linear programming model for nurse scheduling. European Journal Of Operational Research 107(1), 1–18 (1998) 40, 41

[117] Jin, J. : Pré-affectation des tâches aux employés effectuant des tâches non-interruptibles et des activités interruptibles. Ph.D. thesis, École Polytechnique de Montréal (2009) 57

[118] Kawanaka, H., Yamamoto, K., Yoshikawa, T., Shinogi, T., Tsuruoka, S. : Genetic algorithm with the constraints for nurse scheduling problem. In : Proceedings of the 2001 Congress on Evolutionary Computation. vol. 2, pp. 1123–1130. IEEE (2001) 40, 41

[119] Kharraziha, H., Ozana, M., Spjuth, S. : Large scale crew rostering. Carmen Research and Technology Report CRTR-0305, Carmen Systems AB, Gothenburg, Sweden (2003) 47

[120] Knighton, S.A. : A network-based mathematical programming approach to optimal rostering of continuous heterogeneous workforces. Tech. rep., DTIC Document (2005) 38, 43, 57

[121] Kohl, N., Karisch, S.E. : Airline Crew Rostering : Problem Types, Modeling, and Optimization. Annals of Operations Research 127, 223–257 (2004) 39, 47, 49

[122] Kolen, A.W.J., Kroon, L.G. : On the computational complexity of (maximum) class scheduling. European Journal Of Operational Research 54, 23–38 (1991) 51, 52, 57

[123] Kolen, A.W.J., Kroon, L.G. : License class design : complexity and algorithms. European Journal Of Operational Research 63, 432–444 (1992) 51, 54, 57

[124] Kolen, A.W.J., Kroon, L.G. : An analysis of shift class design problems. European Journal Of Operational Research 79, 417–430 (1994) 54, 57

[125] Kolen, A.W.J., Lenstra, J.K., Papadimitriou, C.H., Spieksma, F.C.R. : Interval Scheduling : A Survey. Naval Research Logistics 54, 530–543 (2007) 51

[126] Kovalyov, M.Y., Ng, C.T., Cheng, T.C.E. : Fixed interval scheduling : Models, applications, computational complexity and algorithms. European Journal Of Operational Research 178, 331–342 (2007) 51

[127] Krishnamoorthy, M., Ernst, A.T., Baatar, D. : Algorithms for Large Scale Shift Minimisation Personnel Task Scheduling Problems. European Journal Of Operational Research 219(1), 34–48 (2012) 51, 54, 57, 146, 162, 164

[128] Krishnamoorthy, M., Ernst, A. : The personnel task scheduling problem. In : Yang, X., Teo, K., Caccetta, L. (eds.) Optimization Methods and Applications, Applied Optimization, vol. 52, pp. 343–368. Springer US (2001) 51, 156

[129] Kroon, L.G., Salomon, M., Wassenhove, L.N.V. : Exact and approximation algorithms for the operational fixed interval scheduling problem. European Journal Of Operational Research 82, 190–205 (1995) 52, 57

[130] Kroon, L.G., Salomon, M., Wassenhove, L.N.V. : Exact and approximation algorithms for the tactical fixed interval scheduling problem. Operations Research 45(4) (1997) 54, 57

[131] Kuhn, H.W. : The hungarian method for the assignment problem. Naval Research Logistics Quarterly 2(1-2), 83–97 (1955) 81

[132] Kyngas, N., Nurmi, K., Kyngas, J. : Solving the person-based multitask shift generation problem with breaks. In : Modeling, Simulation and Applied Optimization (ICMSAO), 2013 5th International Conference on. pp. 1–8. IEEE (2013) 57

[133] Lapègue, T., Bellenguez-Morineau, O., Prot, D. : A constraint-based approach for the Shift Design Personnel Task Scheduling Problem with Equity. Computers & Operations Research 40(10), 2450–2465 (2013) 106

[134] Laporte, G. : The art and science of designing rotating schedules. Journal of the Operational Research Society pp. 1011–1017 (1999) 43

[135] Lau, H.C., Lua, S.C. : Efficient Multi-Skill Crew Rostering via Constrained Sets. In : In Proceedings of the Second ILOG Solver and Scheduler Users Conference. pp. 383–396 (1997) 47

[136] Lecoutre, C., Sais, L., Tabary, S., Vidal, V. : Reasoning from last conflict(s) in constraint programming. Artificial Intelligence 173(18), 1592–1614 (2009) 115, 154

[137] Lequy, Q., Bouchard, M., Desaulniers, G., Soumis, F., Tachefine, B. : Assigning multiple activities to work shifts. Journal of Scheduling 15(2), 239–251 (2012) 57

[138] Lequy, Q., Desaulniers, G., Solomon, M.M. : A two-stage heuristic for multi-activity and task assignment to work shifts. Computers & Industrial Engineering 63(4), 831–841 (2012) 57, 78

[139] Li, H., Lim, A., Rodrigues, B. : A hybrid AI approach for nurse rostering problem. In : Proceedings of the 2003 ACM symposium on Applied computing. pp. 730–735. ACM (2003) 40, 96

[140] Lin, S.W., Ying, K.C. : Minimizing shifts for personnel task scheduling problems : A three-phase algorithm. European Journal of Operational Research 237(1), 323 – 334 (2014) 51, 54, 57, 162, 164

[141] Loucks, J.S., Jacobs, R.F. : Tour Scheduling and Task Assignment of a Heterogeneous Work Force : A Heuristic Approach. Decision Sciences 22(4), 719–738 (1991) 43, 44, 57

[142] Lourenço, H.R., Paixão, J.P., Portugal, R. : Multiobjective metaheuristics for the bus driver scheduling problem. Transportation Science 35(3), 331–343 (2001) 39

[143] Love, R.R., Hoey, J.M. : Management science improves Fast-Food operations. Interfaces 20(2), 21–29 (1990) 43, 44, 57

[144] Lučic, P., Teodorovic, D. : Simulated annealing for the multi-objective aircrew rostering problem. Transportation Research Part A : Policy and Practice 33(1), 19–45 (1999) 47, 57

[145] Mabert, V.A., Watts, C.A. : A Simulation Analysis of Tour-Shift Construction Procedures. Management Science 28(5), 520–532 (1982) 42

[146] Maenhout, B., Vanhoucke, M. : A hybrid scatter search heuristic for personalized crew rostering in the airline industry. European Journal of Operational Research 206(1), 155–167 (2010) 47, 57

[147] Martello, S., Toth, P. : A heuristic approach to the bus driver scheduling problem. European Journal of Operational Research 24(1), 106–117 (1986) 39

[148] Mason, A.J., Ryan, D.M., Panton, D.M. : Integrated simulation, heuristic and optimisation approaches to staff scheduling. Operations Research 46(2), 161–175 (1998) 44, 57

[149] Meisels, A., Schaerf, A. : Modelling and solving employee timetabling problems. Annals of Mathematics and Artificial Intelligence 39, 41–59 (2003) 39, 57

[150] Menana, J., Demassey, S. : Weighted Automata, Constraint Programming, and Large Neighborhood Search for the PATAT 2010–NRP Competition (Technical Report) 96

[151] Menana, J., Demassey, S. : Sequencing and counting with the multicost-regular constraint. In : Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, pp. 178–192. Springer (2009) 96

[152] Métivier, J.P., Boizumault, P., Loudni, S. : Solving nurse rostering problems using soft global constraints. In : Principles and Practice of Constraint Programming-CP 2009, pp. 73–87. Springer (2009) 96

[153] Mooney, E., Davidson, T. : Tour scheduling with skill based costs. In : 5th International Conference on Practice and Theory of Automated Timetabling, Pittsburgh, USA (2004) 43, 57

[154] Moudani, W.E., Nunes Cosenza, C.A., de Coligny, M., Mora-Camino, F. : A Bi-Criterion Approach for the Airlines Crew Rostering Problem. In : First International Conference on Evolutionary Multi- Criterion Optimization, Lecture Notes in Computer Science. pp. 486–500 (2001) 47, 49

[155] Moz, M., Pato, M.V. : Solving the problem of rerostering nurse schedules with hard constraints : new multicommodity flow models. Annals of Operations Research 128(1-4), 179–197 (2004) 40

[156] Moz, M., Vazpato, M. : A genetic algorithm approach to a nurse rerostering problem. Computers & Operations Research 34(3), 667–691 (2007) 40

[157] NICTA, O.R.G. : Minizinc challenge 2013 results, http://www.minizinc.org/challenge2013/results2013.html 105

[158] Nobert, Y., Roy, J. : Freight Handling Personnel Scheduling at Air Cargo Terminals. Transportation Science 32(3), 295–301 (1998) 44, 57

[159] Okada, M., Okada, M. : Prolog-based system for nursing staff scheduling implemented on a personal computer. Computers and biomedical research, an international journal 21(1), 53–63 (1988) 40, 57

[160] Ouelhadj, D., Martin, S., Smet, P., Ozcan, E., Vanden Berghe, G. : Fairness in nurse rostering. Technical Report (2012) 50

[161] Ouelhadj, D., Petrovic, S. : A survey of dynamic scheduling in manufacturing systems. Journal of Scheduling 12(4), 417–431 (2009) 169

[162] Pachet, F., Roy, P. : Automatic generation of music programs. In : CP. pp. 331–345 (1999) 148

[163] Palpant, M., Artigues, C., Michelon, P. : LSSPER : Solving the Resource-Constrained Project. Annals of Operations Research 131, 237–257 (2004) 108

[164] Perron, L., Shaw, P., Furnon, V. : Propagation guided large neighborhood search. In : Principles and Practice of Constraint Programming–CP 2004, pp. 468–481. Springer (2004) 122

[165] Pesant, G. : A Constraint Programming Approach to the Traveling Tournament Problem with Predefined Venues. In : Practice and Theory of Automated Timetabling. pp. 303–316 (2012) 109

[166] Pisinger, D., Ropke, S. : Large neighborhood search. In : Gendreau, M., Potvin, J.Y. (eds.) Handbook of Metaheuristics. Forthcoming, 2nd edn. (2009) 108, 122

[167] Prot, D., Lapègue, T., Bellenguez-Morineau, O. : Shift Design for Personel Task Scheduling Problem with Equity (SDPTSP-E) instances (2012), https://sites.google.com/site/sdptspe/ 33

[168] Prot, D., Lapègue, T., Bellenguez-Morineau : A two phase method for the Shift Design Personnel Task Scheduling Problem with Equity objective. (soumis à) International Journal of Production Research (2014) 94

[169] Prud'homme, C., Lorca, X., Jussien, N. : Explanation-guided large neighborhood search. CP Doctoral Program 2013 p. 25 122

[170] Qu, R., He, F. : A hybrid constraint programming approach for nurse rostering problems. In : Applications and innovations in intelligent systems XVI, pp. 211–224. Springer (2009) 40

[171] Quimper, C.G., Rousseau, L.M. : A large neighbourhood search approach to the multi-activity shift scheduling problem. Journal of Heuristics 16(3), 373–392 (2010) 38

[172] Rezanova, N.J., Ryan, D.M. : The train driver recovery problem - a set partitioning based model and solution method. Computers & Operations Research 37(5), 845–856 (2010) 39

[173] Ritzman, L.P., Krajewski, L.J., Showalter, M.J. : Disaggregation of aggregate manpower plans. Management Science 22(11), 1204–1214 (1976) 43

[174] Rong, A. : Monthly tour scheduling models with mixed skills considering weekend off requirements. Computers & Industrial Engineering 59(2), 334–343 (2010) 43, 57

[175] Rossi, A., Singh, A., Sevaux, M. : A metaheuristic for the fixed job scheduling problem under spread time constraints. Computers & Operations Research 37(6), 1045–1054 (2010) 52, 57

[176] Rossi, F., Van Beek, P., Walsh, T. : Handbook of constraint programming. Elsevier (2006) 96

[177] Schaerf, A., Meisels, A. : Solving employee timetabling problems by generalized local search. In : AI* IA 99 : Advances in Artificial Intelligence, pp. 380–389. Springer (2000) 39

[178] Schindler, S., Semmel, T. : Station Staffing at Pan American World Airways. Interfaces 23(3), 91–98 (1993) 44, 57

[179] Schulte, C., Stuckey, P.J. : Efficient constraint propagation engines. ACM Transactions on Programming Languages and Systems 31(1), 1–43 (2008) 152

[180] Shaw, P. : Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. In : Principles and Practice of Constraint Programming (CP 1998), volume 1520 of LNCS. pp. 417–431 (1998) 108, 109

[181] Shebalov, S., Klabjan, D. : Robust airline crew pairing : Move-up crews. Transportation Science 40(3), 300–312 (2006) 39, 47

[182] Smet, P., Vanden Berghe, G. : A matheuristic approach to the shift minimisation personnel task scheduling problem. In : Practice and Theory of Automated Timetabling. pp. 145–151 (2012) 54, 57

[183] Smet, P., Wauters, T., Mihaylov, M., Vanden Berghe, G. : The shift minimisation personnel task scheduling problem : A new hybrid approach and computational insights. Omega 46(0), 64 – 73 (2014) 54, 57

[184] Smet, P., Wauters, T., Mihaylow, M., Vanden Berghe, G. : The shift minimisation personnel task scheduling problem : a new hybrid approach and computational insights (2013), Technical report 156, 162, 164

[185] Smith, S.F. : Reactive scheduling systems. In : Intelligent scheduling systems, pp. 155–192. Springer (1995) 169

[186] Topaloglu, S., Selim, H. : Nurse scheduling using fuzzy modeling approach. Fuzzy Sets and Systems 161(11), 1543–1563 (2010) 40

[187] Türsel Eliiyi, D., Azizoğlu, M. : Heuristics for operational fixed job scheduling problems with working and spread time constraints. International Journal of Production Economics 132(1), 107–121 (2011) 52, 57

[188] Valouxis, C., Gogos, C., Goulas, G., Alefragis, P., Housos, E. : A systematic two phase approach for the nurse rostering problem. European Journal of Operational Research 219(2), 425–433 (2012) 78

[189] Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., De Boeck, L. : Personnel scheduling : A literature review. European Journal of Operational Research 226(3), 367–385 (2013) 38, 157

[190] Vance, P.H., Atamturk, A., Barnhart, C., Gelman, E., Johnson, E.L., Krishna, A., Mahidhara, D., Nemhauser, G.L., Rebello, R. : A heuristic branch-and-price approach for the airline crew pairing problem. (Technical Report) (1997) 39, 47

[191] Veldhoven, S., Post, G., Veen, E., Curtois, T. : An assessment of a days off decomposition approach to personnel scheduling. University of Twente, Department of Applied Mathematics (2013) 38

[192] Wallace, M. : Practical applications of constraint programming. Constraints 1(1-2), 139–168 (1996) 96

[193] Wang, S., Zheng, L., Zhang, Z. : Decomposition Algorithms for the Interval Scheduling Problem. Asia-Pacific Journal of Operational Research 27(04), 517–537 (2010) 52, 57

[194] Weil, G., Heus, K., Patrice, F., Poujade, M. : Constraint Programming for Nurse Scheduling. Engineering in Medicine and Biology Magazine, IEEE 14(4), 417–422 (1995) 40, 96

[195] Wren, A., Rousseau, J.M. : Bus driver scheduling - an overview. Daduna, J. and Branco, I. and and Paixao, J. editors, Springer (1995) 39

[196] Yunes, T.H., Moura, A.V., De Souza, C.C. : Hybrid column generation approaches for urban transit crew management problems. Transportation Science 39(2), 273–288 (2005) 57

l'université nantes angers le mans
COMMUNAUTÉ D'UNIVERSITÉS ET ÉTABLISSEMENTS

MINES Nantes
Institut Mines-Télécom

Thèse de Doctorat

Tanguy LAPÈGUE

Planification de personnel avec affectation de tâches fixées : méthodes et application dans un contexte médical

Workforce scheduling with fixed tasks : methods and application in a medical context

Résumé

Bien que la gestion des ressources humaines soit une problématique bien étudiée, elle reste d'actualité encore aujourd'hui, notamment en raison de la grande diversité des contextes applicatifs. De plus, les outils d'aide à la décision adressant ces problèmes peuvent encore être améliorés. Dans cette thèse, nous nous intéressons au contexte particulier où les activités des employés correspondent à des tâches fixées dans le temps, requérant des compétences précises et ne pouvant être préemptées.
Nous étudions tout d'abord un problème issu de l'industrie pharmaceutique où il s'agit non seulement de trouver une affectation équitable des tâches, mais également de construire les horaires de travail du personnel de manière à respecter les contraintes légales et organisationnelles. Pour résoudre ce problème, nous proposons et comparons deux méthodes exactes, l'une fondée sur un modèle PLNE, l'autre sur un modèle PPC, ainsi que deux méta-heuristiques, l'une reposant sur une décomposition du problème, l'autre reposant sur une recherche par voisinages larges. Nous discutons ensuite de l'intégration de la meilleure de ces méthodes au sein d'un outil d'aide à la décision.
Nous nous intéressons ensuite à un problème d'affectation de tâches fixées visant à minimiser le nombre d'employés requis. Pour résoudre ce problème, nous proposons une approche PPC tirant parti de la structure du problème. Nous montrons que cette approche permet d'obtenir rapidement de bonnes bornes, permettant ainsi de prouver l'optimalité sur les instances de la littérature dans un temps raisonnable.

Mots clés

Planification de personnel avec affectation de tâches fixées, méthodes exactes et approchées, optimisation.

Abstract

Although workforce scheduling has been studied for decades, it remains highly relevant. In particular, applications and contexts are numerous, and they are always changing. Moreover, decision-support tools dedicated to workforce scheduling could still be improved In this thesis, we focus on the particular context where workers are assigned to fixed tasks which cannot be preempted and require specific skills.
First, we study a problem arising in a company specialized in drug evaluation where fixed tasks have to be assigned in a fair way to qualified employees so that the resulting individual shifts and plannings respect legal and organizational constraints. To handle this problem, we design and compare two exact methods, one based on a MIP model, the other on a CP approach, but also two meta-heuristics, one based on a decomposition strategy, and the other on a large neighborhood search. The best method is integrated within a decision support system.
Second, we study a fixed tasks assignment problem, where the objective is to minimize the number of resources required to perform all the tasks. To tackle this problem, we suggest a constraint based approach which takes advantage of the structure of the problem. This approach enables to compute good bounds quickly, and thus, it proves optimality over state-of-the-art instances within a reasonable time limit.

Key Words

Workforce scheduling with fixed tasks, exact and heuristic approaches, optimization.